NCT System

Submitted by: Alexander Nagel

T-00189485

Software Engineering – Stage 2

Date Submitted: 09/12/2016

**Table of Contents**

[1. Introduction/overview 4](#_Toc469045742)

[2. Functional Components 5](#_Toc469045743)

[3. User Requirements 6](#_Toc469045744)

[4. System Requirements 7](#_Toc469045745)

[4.1. System Level Use Case Diagram 7](#_Toc469045746)

[4.2. Manage Registration 8](#_Toc469045747)

[4.2.1. Register Car 8](#_Toc469045748)

[4.2.2. Transfer Ownership 11](#_Toc469045749)

[4.2.3. De-Register Car 13](#_Toc469045750)

[4.2.4. Generate History 15](#_Toc469045751)

[4.3. Manage Centres 17](#_Toc469045752)

[4.3.1. Register Centre 17](#_Toc469045753)

[4.3.2. De-Register Centre 19](#_Toc469045754)

[4.4. Manage Appointments 21](#_Toc469045755)

[4.4.1. Issue Notice 21](#_Toc469045756)

[4.4.2. Make Appointment 23](#_Toc469045757)

[4.4.3. Cancel Appointment 25](#_Toc469045758)

[4.4.4. Record Results 27](#_Toc469045759)

[4.4.5. List Appointments 27](#_Toc469045760)

[5. System Model 29](#_Toc469045761)

[5.1. Level-0 DFD 30](#_Toc469045762)

[5.2. Level-1 DFD 30](#_Toc469045763)

[5.3. Level-2 DFD (Process P1: Manage Cars) 31](#_Toc469045764)

[5.4. Level-2 DFD (Process P2: Manage Centres) 32](#_Toc469045765)

[5.5. Level-2 DFD (Process P3: Manage Appointments) 32](#_Toc469045766)

[6. Data Model (Class Diagram) 34](#_Toc469045767)

[6.1. Class Diagram 34](#_Toc469045768)

[6.2. Relational Schema 34](#_Toc469045769)

[6.3. Database Schema 35](#_Toc469045770)

[7. Conclusion 36](#_Toc469045771)

[8. Appendices 37](#_Toc469045772)

[8.1. Appendix A – Title 37](#_Toc469045773)

[8.2. Appendix B – Title 37](#_Toc469045774)

# Introduction/overview

I considered to create my Software Engineering project “NCTSYS” which is partly based on a National Car Test Services structure. The System requires interactivity with a Primary Business Actor **‘Operator’** and two Participating Actors, Owner of the Vehicle and Test Centre Mechanic.

The System provides system administration tools like Register Vehicle, Register New Test Centre etc. implements by the Operator and Car Test related tools like Record Test Results implements by a Mechanic and Appointment related tools like Make Appointment or Cancel Appointment and require an Owner of the Vehicle or the Operator interaction.

# Functional Components

A ***hierarchy chart*** representing the functional components of NCT System.

# User Requirements

High level **abstract statements** describing the user requirements.

**3.1 NCTSYS will manage for car registration.**

3.1.1. NCTSYS will register/transfer a car ownership

3.1.2. NCTSYS will de-register a car

3.1.3. NCTSYS will generate a car history

**3.2. NCTSYS will Test Centre administration.**

3.2.1. NCTSYS will register a new Test Centre

3.2.2. NCTSYS will de-register a Test Centre

**3.3. NCTSYS will manage appointments.**

3.3.1. NCTSYS will Issue NCT Notices

3.3.2. NCTSYS will allow an appointment to be made

3.3.3. NCTSYS will allow an appointment to be cancelled

3.3.4. NCTSYS will record test results

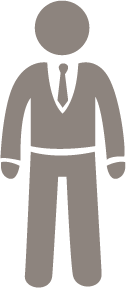
3.3.5. NCTSYS will generate list of Daily appointments

# System Requirements

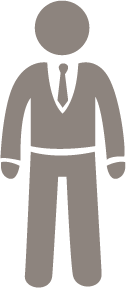
The System requirements are outlined below using use case modelling. Each use case narrative explains a specific requirements of the system.

## System Level Use Case Diagram

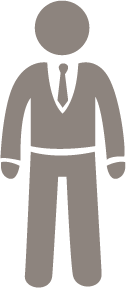
This is a high level description of the functions needed in a library system



Operator



Owner

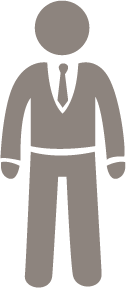


Mechanic

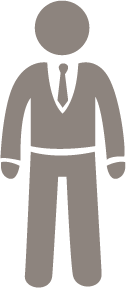
## Manage Registration

### Register/Transfer a Car Ownership

This function will allow to register/transfer a vehicle ownership.



Owner



Operator

<<includes>>

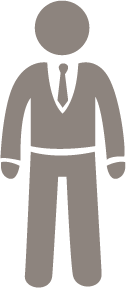
<<Extends>>

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Register/Transfer Ownership | |
|  |  | |
| **Use Case Id** | 4.2.1 | |
| **Priority** | High | |
| **Source** | Owner | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | Owner | |
| **Description** | This function Registers or Transfers a car ownership on the system. | |
| **Preconditions** | Owner must complete and sign a NCT Registration form. | |
| **Trigger** | None | |
| **Expected Scenario** | **Operator** | **System Response** |
|  | **Step 1:** Operator invokes Register Car function  **Step 3:** Operator enters Car details:   * Registration Number   **Step 5:** Operator selects Car details:   * Make * Model * Engine Sizes * Colour * Fuel Type * First Registration Date   **Step 6:** Operator submits Car details  **Step 9:** Operator enters PPSN  **Step 12:** Operator enters Owner details:   * Surname * Forename * DOB * Tel * E-mail * Address1 * Address2 * County   **Step 13:** Operator presses the Register Button to register details | **Step 2:** System loads lists of Makes, Makes, Colours, Fuel Types Engine Sizes, and Counties and Display UI  **Step 4:** System validates Registration Number and checks if Registration Number already exists in Cars file   * **If exists:** system loads car details on UI for viewing only and goes to **step 9**   **If not exists:** go to **Step 5**  **Step 7:** System validates details:   * All fields must be entered * First Registration date must not be in the future   **Step 8: System prompts user to enter PPSN**  **Step 10:** System validates PPSN format  **Step 11:** System checks if PPSN already exists in Owners File   * **If exists:** system loads owner details on UI for viewing only and goes to step 15 * **If not exists:** go to **Step 12**   **Step 14: if not existing owner,** System validates details:   * All fields must be entered * Text only values must not contain numeric * Registration Number must be in valid format * Dates cannot be in the future * PPSN must be in valid format * E-mail must be in valid format * Telephone Number must contain only numbers * Owner must be at least 17 years’ old   **Step 15:** Car is assigned a default status as **‘active’**  **If Car exists:** system goes to **step 17**  **If Car not exists:** system goes to **step 16**  **Step 16: If Car exists:** system goes to **step 17**  **If Car not exists:** SystemSaves Car details in **Car File**   * Registration Number * Make * Model * Engine Sizes * Fuel Type * Colour * First Registration Date * Status   **Step 17: If Owner exists:** system goes to **step 18**  **If Owner not exists:** System saves Owner details in **Owner File**   * PPSN * Surname * Forename * DOB * Tel * E-mail * Address1 * Address2 * County   **Step 18:** Save Car Registration details in **Registrations File**   * Registration Date   *Current System date*   * PPSN * Registration Number   **Step 19**: Display confirmation message  **Step 20:** Clear the UI |
| **Alternate Scenarios** | **Operator** | **System Response** |
| **Invalid Data Entered** |  | **Step 10:** Invalid Data detected  **Step 11:** Display appropriate error message and return to **step 9** |
| **Conclusions** | Car is now registered and approved to enter appointment section | |
| **Post conditions** |  | |
| **Business Rules** | Car must be registered in Driver and Vehicle Computer Services Division, Department of Transport and Owner must be at least 17 years old.  Car ownership cannot be transferred on the day of the registration. | |
| **Implementation Constraints** |  | |

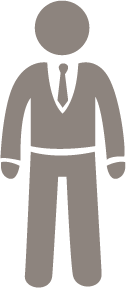
### De-Register Car

This function changes a Car status from Active to Inactive

When a Car is Scrapped or Off the Road, the Car is Deregistered



Owner

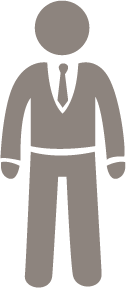


Operator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | De-Register Car | |
|  |  | |
| **Use Case Id** | 4.2.2 | |
| **Priority** | Low | |
| **Source** | Owner | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | Owner | |
| **Description** | This function changes a Car status from Active to Inactive | |
| **Preconditions** | Proof of fatal damage or proof of export of the vehicle must be provided . | |
| **Trigger** | None | |
| **Expected Scenario** | **Operator** | **System Response** |
|  | **Step 1:** Operator invokes the de-register Car function  **Step 3:** Operator enters Registration Number  **Step 6**: Operator clicks on Continue button  **Step 8**: Operator clicks on De-Register button | **Step 2:** Display UI  **Step 4:** System validates details:   * Registration Number   **Step 5: If Car Inactive:** System displays Warning message and goes to **step 3**  **If Car Active:** System retrieves appropriate Car details from **Car file** and displays on UI  **Step 7:** System loads De-Register button  **Step 9:** Systemupdates Car status to Inactive in **Car file:**  **Step 10:** System displays confirmation message  **Step 11:** System clears UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 5:** Invalid Data detected  **Step 6:** Display appropriate error message and return to **Step 3** |
| **Conclusions** | Car is now de-registered and not allowed to enter appointment section and no longer on NCT Notice list | |
| **Post conditions** |  | |
| **Business Rules** | Only vehicles with active status may be de-registered | |
| **Implementation Constraints** |  | |

### Generate History

This function will generate Ownership history table and send a file to printing device for a selected Car



Operator

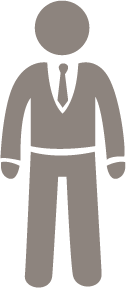
|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Generate History | |
|  |  | |
| **Use Case Id** | 4.2.4 | |
| **Priority** | Low | |
| **Source** | NCT System | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | None | |
| **Description** | This function will generate Ownership history for a selected Car | |
| **Preconditions** | None | |
| **Trigger** | None | |
| **Expected Scenario** | **Operator** | **System Response** |
|  | **Step 1:** Operator invokes the Generate History function  **Step 3:** Operator enters Registration Number  **Step 6:** Operator exits from Generate History function | **Step 2:** Display UI  **Step 4:** System validates details:   * Registration Number must exist within the System   **Step 5:** System retrieves details of appropriate Car Owners from **Owners file**, **CarReg file**, **Car file** and generates History table in Date descending order and displays on UI  **Step 9:** System displays confirmation message  **Step 10:** System clears UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** | **Step 6**: Operator print a file to printing device | **Step 4:** Invalid Data detected  **Step 5:** Display appropriate error message and return to **Step 3**  **Step 7**: System generates Print file and sends it to corresponding device  **Step 8:** System displays confirmation message  **Step 9:** System clears UI |
| **Conclusions** | The Car Ownership History is available on hard copy | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Ownership History 02-C-23765 | | | | | |
| DL ID | Date | Surname | Forename | e-mail | County |
| 376632767 | 23/04/2016 | Bill | Gates | bill.gates@gm.com | Dublin |
| 982389928 | 23/04/2014 | Mark | Zuckerberg | mark.zuck@gm.com | Galway |
| 675654322 | 23/04/2007 | John | Travolta | john.travolta@gm.com | Kerry |
| 983988333 | 23/04/2002 | Steve | Jobs | steve.jobs@gm.com | Cork |

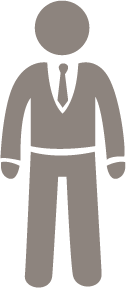
## Manage Centres

### Register Centre

This function will allow to register a new Test Centre.



Owner



Operator

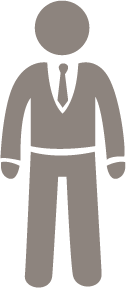
<<includes>>

<<Extends>>

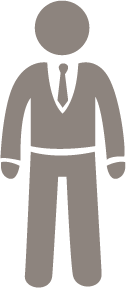
|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Register Centre | |
|  |  | |
| **Use Case Id** | 4.3.1 | |
| **Priority** | High | |
| **Source** | Test Centre Owner | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | Test Centre Owner | |
| **Description** | This function Registers a New Test Centres details on the system. | |
| **Preconditions** | Test Centre Owner must complete and sign a Test Centre Registration form. | |
| **Trigger** | None | |
| **Expected Scenario** | **Operator** | **System Response** |
|  | **Step 1:** Operator invokes Register Centre function  **Step 3:** Operator enters Centre details:   * Centre Name * Address 1 * Address 2 * County * Tel * E-mail * Number of Staff * Date | **Step 2:** System retrieve list of Counties from Counties table  **Step 4:** System validates details:   * All fields must be entered * Text only values must not contain numeric * Date cannot be in the future * E-mail must be in valid format   **Step 5:** Assign Centre ID  **Step 6:** Centre is assigned a default status of **‘active’**  **Step 7:** Save Centre details in **Centres File**  **Step 8**: Display confirmation message  **Step 9:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 4:** Invalid Data detected  **Step 5:** Display appropriate error message and return to **Step 3** |
| **Conclusions** | Centre is now registered and available for use | |
| **Post conditions** |  | |
| **Business Rules** | All Staff Members must hold valid SOS mechanic certificate | |
| **Implementation Constraints** |  | |

### De-Register Centre

This function will allow to close a Test Centre.



Operator

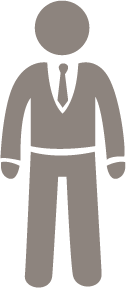


Owner

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | De-Register Centre | |
|  |  | |
| **Use Case Id** | 4.3.2 | |
| **Priority** | High | |
| **Source** | Test Centre Owner | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | Test Centre Owner | |
| **Description** | This function De-Registers a Test Centres and sets status to ‘Inactive’. | |
| **Preconditions** | Owner must complete and sign a Test Centre De-Registration form.  Test Centre must exist within the System and have a status **‘active’**. | |
| **Trigger** | None | |
| **Expected Scenario** | **Operator** | **System Response** |
|  | **Step 1:** Operator invokes De-Register Centre function  **Step 3:** Operator selects County from a list  **Step 5:** Operator selects a designated Test Centre  **Step 7:** Operator prompts to trigger De-Registration function | **Step 2:** System retrieve list of Counties from Counties table and Display UI  **Step 4:** System retrieve list of appropriate Test Centres  **Step 6:** System retrieve summary details of appropriate Test Centre and Display on UI  **Step 8:** Centre is assigned a status of **‘inactive’**  **Step 7:** Save Centre details in **Centres File**  **Step 8**: Display confirmation message  **Step 9:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  |  |
| **Conclusions** | Centre is now Deregistered and not available for use | |
| **Post conditions** | No appointments can by made at this centre | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

## Manage Appointments

### Issue Notice

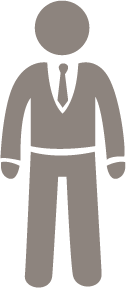


Operator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Issue Notice | |
|  |  | |
| **Use Case Id** | 4.4.1 | |
| **Priority** | High | |
| **Source** |  | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | None | |
| **Description** | This function Issue NCT Notices to Car Owners | |
| **Preconditions** | Owner must complete and sign a Test Centre De-Registration form.  Test Centre must exist within the System and have a status **‘active’**. | |
| **Trigger** | None | |
| **Expected Scenario** | **Operator** | **System Response** |
|  | **Step 1:** Operator invokes Issue Notice function  **Step 3:** Operator initiates Issue Notice function | **Step 2:** System retrieve list of Registration Numbers which never been tested but 3 or more years old, due in <=28 days, over due from **Notice file** and corresponded Owner details from **Owner file** and Display UI  **Step 4:** System generates and sends e-mail to corresponded Owner with appropriate message for each Registration Number  **Step 8**: Display confirmation message  **Step 9:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  |  |
| **Conclusions** | NCT Notices have been sent | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Make Appointment

This function will allow an appointment to be made.

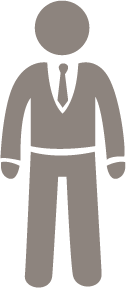


Owner

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Make Appointment | |
|  |  | |
| **Use Case Id** | 4.4.2 | |
| **Priority** | High | |
| **Source** | Owner | |
| **Primary Business Actor** | Owner | |
| **Other Participating Actors** | None | |
| **Description** | This function will allow an appointment to be made. | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Owner** | **System Response** |
|  | **Step 1:** Owner invokes Make Appointment function  **Step 3:** Owner enters Registration Number  **Step 5**: Owner identifies and confirms a Car  **Step 7**: Owner selects Test Centre and date required  **Step 9**: Owner selects required time and submits it | **Step 2:** System Display UI  **Step 4:** System retrieve summary details of appropriate Car from **Car File** and **Owner file** and Displays on UI  **Step 6**: System retrieve list of Test Centres from **Centres File** and Displays on UI  **Step 8**: System retrieve available Time slots for selected Centre and Date from **Appointment File** and Display on UI  **Step 10:** Assign Appointment ID  **Step 11**: System sets   * Status as **‘active’** * Appointment date * Appointment time slot   and saves details in **Appointment File**  **Step 12**: Display confirmation message  **Step 13:** Clear the UI |
| **Alternate Scenarios** | **Owner** | **System Response** |
| **Invalid Data Entered** | **Step 5**: Owner do not identify a Car | **Step 6:** Display appropriate error message and return to **Step 3** |
| **Conclusions** | Appointment is set. | |
| **Post conditions** |  | |
| **Business Rules** | Car must exist within the System and must have **‘active’** status . | |
| **Implementation Constraints** |  | |

### Cancel Appointment

This function will cancel appointment



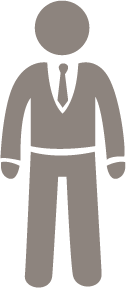
Owner

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Cancel Appointment | |
|  |  | |
| **Use Case Id** | 4.4.3 | |
| **Priority** | High | |
| **Source** | Owner | |
| **Primary Business Actor** | Owner | |
| **Other Participating Actors** | None | |
| **Description** | This function will allow cancel appointment. | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Owner** | **System Response** |
|  | **Step 1:** Owner invokes Cancel Appointment function  **Step 3:** Owner enters Registration Number  **Step 5**: Owner identifies and confirms a Car  **Step 7**: Owner selects Cancel option | **Step 2:** System Display UI  **Step 4:** System retrieve summary details of appropriate Car from **CarReg File** and Displays on UI  **Step 6**: System retrieve summary details of currant Appointment from **Appointment File** and Displays on UI  **Step 8**: System sets Dates and Time slot as ‘available’ and save in to **Schedule File**  **Step 10**: System sets Appointment status as ‘cancelled’ and  saves in **Appointment File**  **Step 8**: Display confirmation message  **Step 9:** Clears UI |
| **Alternate Scenarios** | **Owner** | **System Response** |
| **Invalid Data Entered** | **Step 5**: Owner do not identify a Car | **Step 6:** Display appropriate error message and return to **Step 3** |
| **Conclusions** | Appointment is cancelled. | |
| **Post conditions** |  | |
| **Business Rules** | Car must exist within the System and must have **‘active’** status . | |
| **Implementation Constraints** |  | |

### Record Results

### List Appointments

This function will generate list of Daily appointments.



Operator

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | List Appointments | |
|  |  | |
| **Use Case Id** | 4.4.5 | |
| **Priority** | Low | |
| **Source** | System | |
| **Primary Business Actor** | Operator | |
| **Other Participating Actors** | None | |
| **Description** | This function will generate list of Daily appointments. | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Owner** | **System Response** |
|  | **Step 1:** Operator invokes List Appointments function  **Step 3:** Operator selects appropriate Test Centre  **Step 5**: Operator selects print function | **Step 2:** System retrieve list of Test Centres from **Centres file** and display UI  **Step 4:** System retrieve list appointment from **Appointment file** for current system date and generates Preview table in TIME ascending order and displays on UI  **Step 6**: System generates a print file and sends it to corresponding device  **Step 9:** System displays confirmation message  **Step 10:** System clears UI |
| **Alternate Scenarios** | **Owner** | **System Response** |
| **Invalid Data Entered** |  |  |
| **Conclusions** | Daily Appointment list is displayed and printed. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

# System Model

**Data Flow Diagrams - Elements:**

*External Entities:*

Owner

Mechanic

*Data Stores:*

D1 Car File

D2 Owners File

D3 CarReg File

D4 Centres File

D5 Notice File

D6 Appointment File

D7 TestRecord File

*Processes:*

P1 Manage Cars

P1.1 Register Car

P1.2 Transfer Ownership

P1.3 De-Register Car

P1.4 View Car Owners History

P2 Manage Centres

P2.1 Open Centre

P2.2 Close Centre

P3 Manage Appointments

P3.1 Issue Notice

P3.2 Make Appointment

P3.3 Cancel Appointment

P3.4 Record Result

P3.5 List Daily Appointments

## Level-0 DFD

Car Details

Test Results

NCTSYS

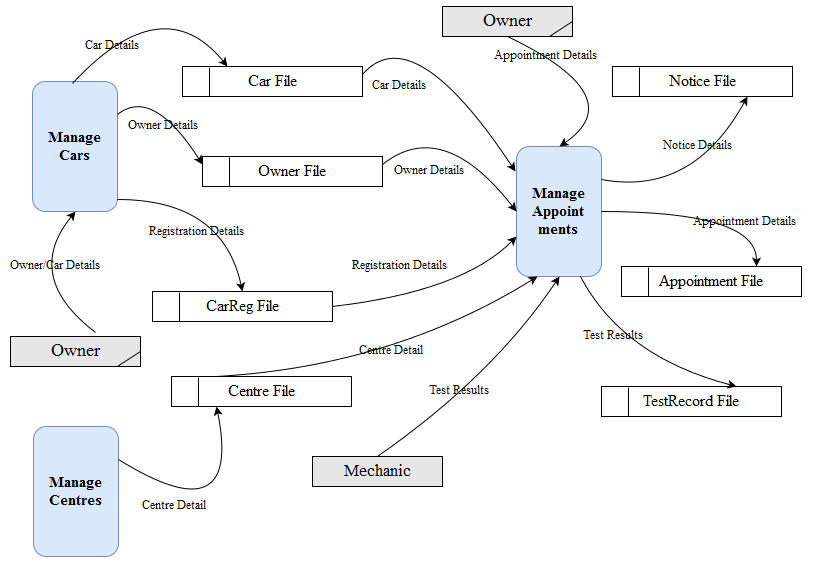
Mechanic

Owner

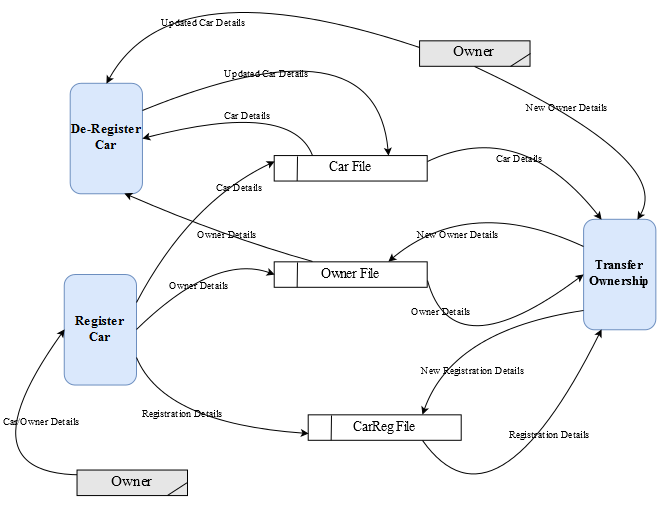
Daily Appointments

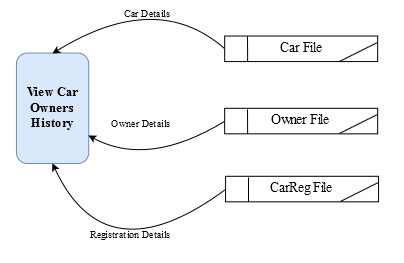
Appointment

## Level-1 DFD

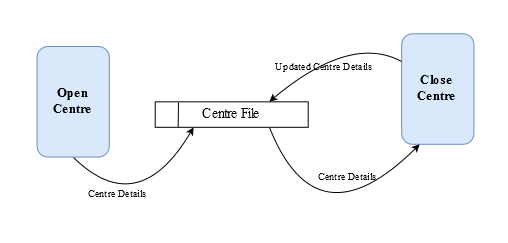


## Level-2 DFD (Process P1: Manage Cars)

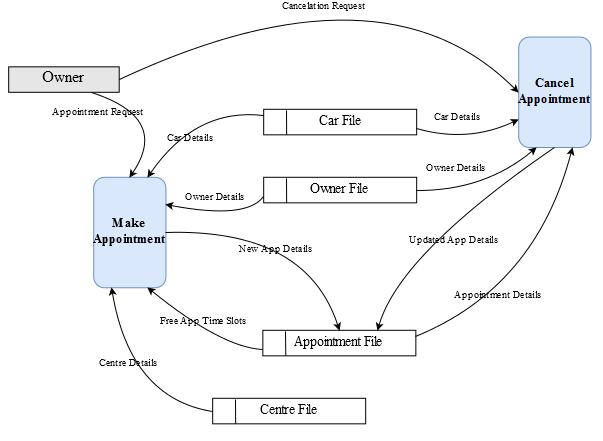


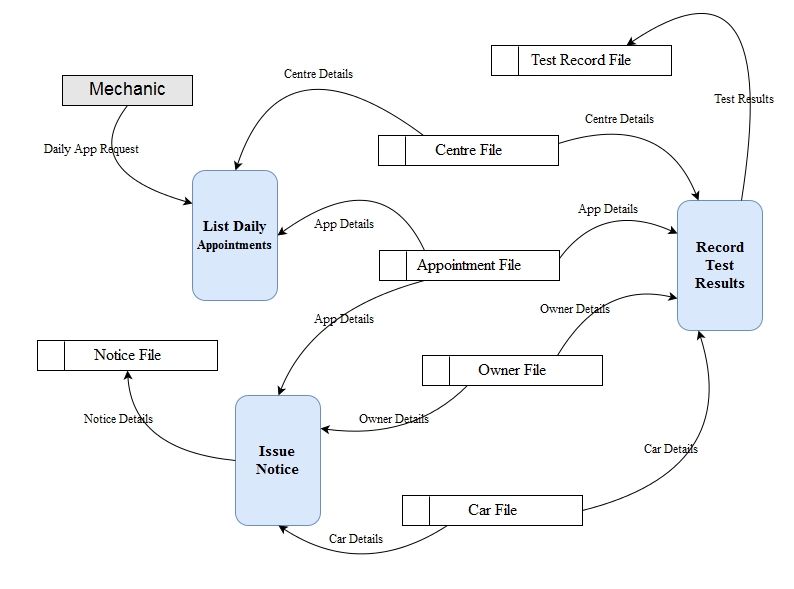


## Level-2 DFD (Process P2: Manage Centres)



## Level-2 DFD (Process P3: Manage Appointments)



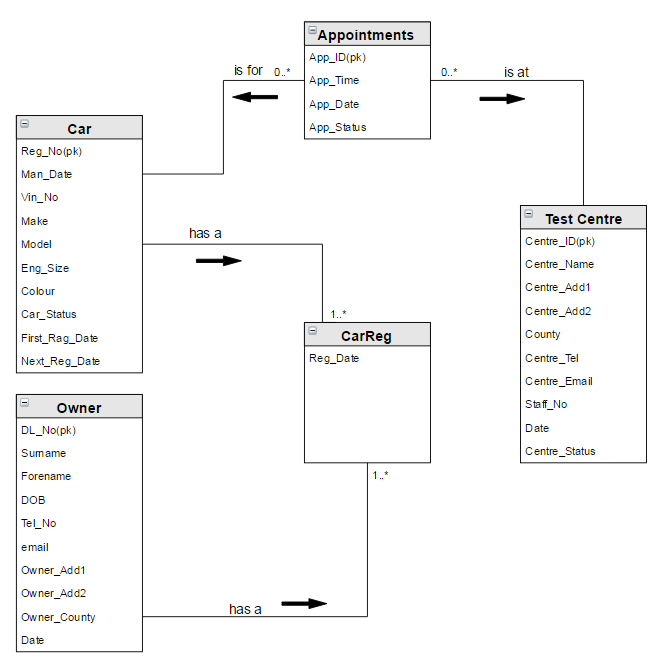


# Data Model (Class Diagram)

## Class Diagram

Object Model – UML Class Diagram

Class diagram shows objects & attributes



## Relational Schema <NCTSYS>

Cars (Reg\_No, Man\_Date, Vin\_No, Make, Model, Eng\_Size, Colour, Car\_Status, First\_Reg\_Date, App\_Date)

Owners (DL\_No, Surname, Forename, DOB, Tel\_No, email, Owner\_Add1, Owner\_Add2, Owner\_County, Date)

CarReg (Reg\_Date, DL\_No, Reg\_No)

Appointments (App\_ID, App\_Time, App\_Date, App\_Status)

Centres (Centre\_ID, App\_ID, Centre\_Name, Centre\_Add1, Centre\_Add2, County, Centre\_Tel, Centre\_Email, Centre\_Status, Date)

## Database Schema

**Relation**: Cars

**Attributes:**

Reg\_No char (10) NOT NULL,

Man\_Date date NOT NULL,

Vin\_No char (17) NOT NULL,

Make varchar (25) NOT NULL,

Model varchar (15) NOT NULL,

Eng\_Size numeric (4) NOT NULL,

Colour varchar (15) NOT NULL,

Car\_Status char (1) DEFAULT ‘A’,

First\_Reg\_Date date NOT NULL,

App\_Date date,

**PRIMARY KEY (Reg\_No)**

**FORGEIN KEY (App\_Date) REFERNCES Appointments**

**Relation**: Owners

**Attributes:**

DL\_No char (10) NOT NULL,

Surname varchar (30) NOT NULL,

Forename varchar (30) NOT NULL,

DOB date NOT NULL,

Tel\_No varchar (10) NOT NULL,

Email varchar (30) NOT NULL,

Owner\_Add1 varchar (20) NOT NULL,

Owner\_Add2 varchar (20) NOT NULL,

Owner\_County varchar (15) NOT NULL,

Date date DEFAULT sysdate,

**PRIMARY KEY (DL\_No)**

**Relation**: Centres

**Attributes:**

Centre\_ID numeric (2) NOT NULL,

App\_ID numeric (5) NOT NULL,

Centre\_Name varchar (15) NOT NULL,

Centre\_Add1 varchar (20) NOT NULL,

Centre\_Add2 varchar (20) NOT NULL,

County varchar (15) NOT NULL,

Centre\_Tel varchar (10) NOT NULL,

Centre\_Email varchar (30) NOT NULL,

Centre\_Status char (1) DEFAULT ‘A’,

Date date NOT NULL DEFAULT sysdate,

**PRIMARY KEY (Centre\_ID)**

**FORGEIN KEY (App\_ID) REFERNCES Appointments**

**Relation**: CarReg

**Attributes:**

Reg\_Date date NOT NULL,

DL\_No char (10) NOT NULL,

Reg\_No char (10) NOT NULL,

**PRIMARY KEY (Reg\_Date)**

**FORGEIN KEY (DL\_No) REFERNCES Owners**

**FORGEIN KEY (Reg\_No) REFERNCES Cars**

**Relation**: Appointments

**Attributes:**

App\_ID numeric (5) NOT NULL,

App\_Time time NOT NULL,

App\_Date date NOT NULL,

App\_Status char (1) DEFAULT ‘M’, for MADE

**PRIMARY KEY (App\_ID)**

# Conclusion

The System will provide all necessary tools to keep Roads safety regarding mechanical state of the vehicle.

The System could provide Nation-wide control of the Vehicles on Irish roads to Department of Transport.

# Appendices

## Appendix A – Title

## Appendix B – Title

Might include:

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