**CRIME MANAGEMENT SYSTEM**

## A MINI PROJECT REPORT

**Submitted in partial fulfillment of the requirements for the award of the degree of**

**BACHELOR OF COMPUTER APPLICATIONS**

**Submitted By D. DIVYA**

**Register No. 19UCA221730**

**Under the Guidance of**

**Ms. A.RAJALAKSHMI, M.Sc., M. Phil**

**Guest Lecturer of Computer Applications**



**DEPARTMENT OF COMPUTER APPLICATIONS GOVERNMENT ARTS COLLEGE (AUTONOMOUS)**

**(Re-Accredited by NAAC with B Status) SALEM-636 007**

**MAY 2022**

**CERTIFICATE**

This is to certify that the project work entitled “**CRIME MANAGEMENT SYSTEM”** submitted to Government Arts College (Autonomous), Salem-7. in partial fulfillment requirements for the award of degree of **BACHELOR OF COMPUTER APPLICATIONS** is a record of project work done by [**D. DIVYA], (Register No : 19UCA221730)** between May 2022 under my supervision and guidance.

Date : **Signature of the Guide**

Place : Salem-7

**Head of the Department**

Submitted for the Autonomous Viva-Voce Examination held on……………………..

### Internal Examiner External Examiner

**DECLARATION**

I here by declare that the mini project work entitled “**CRIME MANAGE MENT SYSTEM**” submitted to Government Arts College (Autonomous), Salem-7 in partial fulfillment of the requirements for the award of degree of **BACHELOR OF COMPUTER APPLICATIONS** is a record of original work done by me under the supervision and guidance of **Ms. A.RAJALAKSHMI, M.Sc., M. Phil** guest lecturer in department of Computer applications, Government Arts College (Autonomous), Salem-7.

Signature of the

Candidate

(**D.DIVYA**)

**REGISTER NUMBER:**

**19UCA221730**

Date:

Place: Salem-7.

**ACKNOWLEDGEMENT**

At the outset, I would like to thank and honor God who gave me the wisdom and knowledge to complete this project.

I would like to profusely thank to our college principal **Dr.S.KALAISELVAN,M.Sc.,M.Phil.,M.Ed.,Ph.D** Government Arts College (Autonomous), Salem-7 for her guidance and encouragement to successfully complete my project.

I express my gratitude **Mrs. A. VIJAYA, MCA, M.Phil, Ph.D** Head of the Department of Computer Applications, Government Arts College (Autonomous), Salem-7 for being an unfailing source of inspiration and encouragement.

I would like to express my gratitude and sincere thanks to my guide of **Ms. A.RAJALAKSHMI, M.Sc., M.Phil** guest lecture**,** Department of Computer Applications, Government Arts College (Autonomous), Salem-7 for her source of guidance and suggestions throughout the project.

I express my sincere thanks to all other staff members of computer science department for this supporting help.

This acknowledgement will be incomplete without expressing my thanks to my parents and friends for their constant support and encouragement.

# CONTENTS

|  |  |  |
| --- | --- | --- |
| **S.NO** | **CONTENTS** | **P.NO** |
|  | **ABSTRACT** |  |
| **1** | **INTRODUCTION** |  |
|  | 1.1 Project Description |  |
| **2** | **SYSTEM ANALYSIS** |  |
| 2.1 Existing System |  |
| 2.1.1 Drawbacks of Existing system |  |
| 2.2 Proposed System |  |
| 2.2.1 Advantages of Proposed System |  |
| 2.3 System Requirement and specification |  |
| 2.3.1 Hardware Requirements |  |
| 2.3.2 Software Requirements |  |
| 2.3.3 Software Description |  |
| **3** | **SYSTEM DESIGN AND**  **DEVELOPMENT** |  |
| 3.1 Data Flow Diagram |  |
| 3.2 Module Description |  |
| 3.3 Input Design |  |
| 3.4 Output Design |  |
| 3.5 Database Design |  |
| 3.6 Code Design |  |
| 3.7 Reports Design |  |
| **4** | **TESTING AND IMPLEMENTATION** |  |
| 4.1 System Testing |  |
| 4.2 System Implementation |  |
| **5** | **CONCLUSION** |  |
| **6** | **FUTURE ENHANCEMENT** |  |
| **7** | **BIBLIOGRAPHY** |  |
| 7.1 Reference Books |  |
| 7.2Website References |  |

**INTRODUCTION**

## PROJECT DESCRIPTION

### CRIME FILE MANAGEMENT SYSTEM ABSTRACT

The objective of our project is to design and develop a database which enables the officials of police department to store and retrieve the details about the case or criminal in an effective manner. Currently all the works are done by hand coding, it is time consuming and difficult to search about crimes and criminals in case of existing system. Searching criminal records through this criminal database is far simplest and quickest way to obtain the required information of any detail about the particular case and the particular detail about any case or criminal. It is possible to obtain the crime or criminal record by simply entering the name of the person, place of occurrence and date of occurrence of crime, FIR number and by the name of investigator who handled that particular case.

Front- End: ASP.NET

Back- End: MSSQL

The Activities Associated with Eight Modules

* + - * Login Page
      * Home Page
      * Investigator Details
      * New User Registration
      * FIR
      * Crime Details
      * Action Taken
      * Search

# SYSTEM ANALYSIS

## SYSTEM ANALYSIS

### 2.1 INTRODUCTION TO SYSTEM ANALYSIS

System analysis is a process of gathering and interpreting facts, diagnosing problem and the information to recommend improvements on the system. It is a problem-solving activity that requires intensive communication between the system users and system developers. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system identified. The outputs from the organizations are traced to the various process. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal.

Preliminary study is problem solving activities that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies, a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

### 2.2 EXISTING SYSTEM:

* Lack of security of data
* More than power
* Time consuming
* Consumes large volume of pare work
* Needs manual calculations.

## 2.3 PROPOSED SYSTEM

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides proper security and reduces the manual work.

* Security of data
* Ensure data accuracy’s
* Minimize manual data entry
* Greater efficiency
* Better service
* User friendliness and interactive
* Minimum time required

## 2.4 SYSTEM REQUIREMENTS

**2.4.1 HARDWARE REQUIREMENTS**

* RAM : 3 GB or more
* Hard disk : 320 GB or more
* Monitor : VGA/SVGA
* Keyboard : 104 keys
* Mouse : 2 buttons / 3 buttons

**2.4.2 SOFTWARE REQUIRENETS**

* Operating system : Windows 10 or above
* Front end : Asp .Net
* Backend : SQL Server

### SOFTWARE DESCRIPTION

**ASP.NET**

Microsoft has built a very powerful framework and most successful web application development framework. Microsoft is delivering many new updates which are new and extended features that help the developers to make [their web application](https://www.educba.com/what-is-web-application/) highly scalable. Also, which leads to high performance. If we coupled both things that are application monitoring with other performance tools like .net profiler tool (used to improve line by line code), it results in a more powerful solution for building an unbelievable application.

To boost up and increase the performance of the application it has many great features that also help to overcome the common development challenges. Web pages and technologies are created with the help of ASP.NET.

**FEATURES IN ASP.NET**

* Dynamic Language Runtime.
* Expanded Base Class.
* Parallel Computing.
* Managed Extensibility Framework.
* Covariance and Contravariance.
* Big Integer and Complex Numbers.

## SQL DATABASE DESIGN

A database is an organized mechanism that has the capability of storing information through which is user can retrieve stored information in an effective and efficient manner.

The database design is a two-level process. In this first step, user requirements are gathered together and a database is designed which will meet these requirements as clearly as possible. This step is called Information Level Design and it is taken independent of any individual DBMS.

In the second step, this information level design is transferred into a design for the specific DBMS that will be used to implement the system in questions. This step is called physical level design, concerned with the characteristics of the specific DBMS that will be used. A database design runs parallel with the system design. The organization of the data in the database is aimed to achieve the following to major objectives.

* Data Integrity
* Data Independence

Normalization is the process of decomposing the attributes in an application, which results in a set of tables with every simple structure. The purpose of normalization is to make tables as simple as possible. Normalization is carried out in the system for the following reason.

* To structure the data so that there is no repetition of data, this helps in saving.
* To permit simple retrieval of data in response to query and report request.
* To simplify the maintenance of the data through updates, insertions, deletions.
* To reduce the need to restructure or recognize data which new application requirements arise.

**RELATIONAL DATABASE MANAGEMENT SYSTEM (RDBMS):**

A relational model represents the database as a collection of relations.

Each relation resembles a data of values or file of records. In formal relational model terminology, a row is called tuple, a column header is called an attribute and the table is called a relation. A relational database consists of a collection of tables, each of which is assigned a unique name. A row in a table represents a set of related values.

**RELATIONSHIPS:**

Table relationships are established using key. The two main keys of prime importance are primary key and foreign key. Entity integrity and referential integrity relationships can be established with the keys. Entity integrity enforce that no primary key can have null values. Referential integrity enforce that no primary key can have null values.

**NORMALIZATION:**

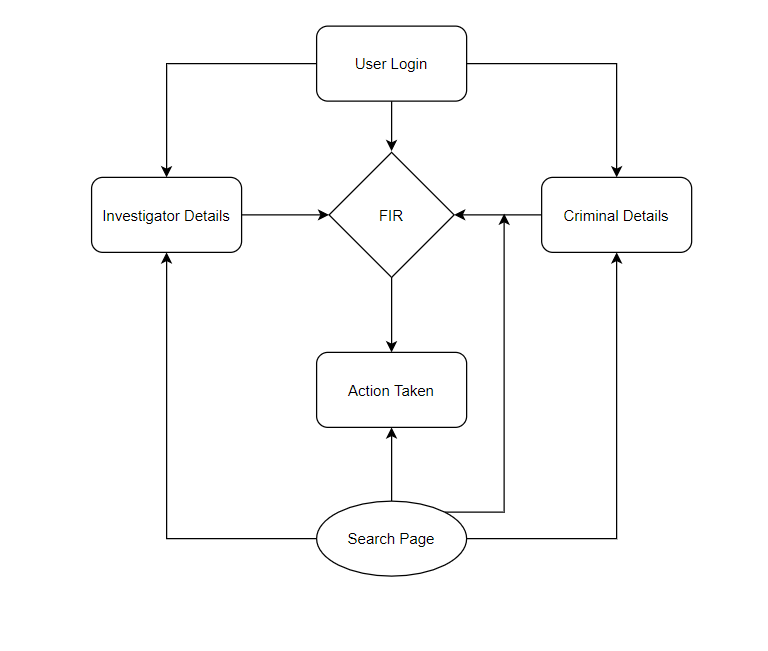
As the name implies, it denoted putting things in the normal form. The application developer via normalization tries to achieve a sensible organization of data into proper tables and columns and where names can be easily correlated to the data by the user. Normalization eliminates repeating groups at data and there by avoids data redundancy which proves to be a great burden on the computer resources. Thus incudes.

* Normalize the data.
* Choose proper names for the tables and columns.
* Choose the proper name for the data.

# SYSTEM DESIGN AND DEVELOPMENT

## 3.1 SYSTEM DESIGN AND DEVELOPMENT

* 1. **DATA FLOW DIAGRAM**



### MODULE DESCRIPTION

The Project having Eight Modules

* + - * Login Page
      * Home Page
      * Investigator Details
      * New User Registration
      * FIR
      * Crime Details
      * Action Taken
      * Search

### Login Page

There will be a username and password to login into the system to use all the facilities.

### Home Page

In this module user has two options to go with they are

* File FIR
* Search details
* Investigator registration
* Logout

## Investigator Details

The investigator is the police officer who investigates the case. There may be more than one Investigator in one police station

### New User Registration

The Registration module enables the user to register himself to the crime file management system.

### FIR

First Information Report (FIR) is a document prepared by the police when they receive information from the petitioner about the commission of a cognizable offence.

### Criminal Details

This module holds the basic information about the criminal enables us to view the status criminal.

### Action Taken

This module has the information about what action taken for the given complaint.

### Search

Here we have the facility to search the details of the criminals and cases. The user can search the crime and criminal’s details by using the name, FIR number, place of occurrence, date of occurrence, type of crime, Investigator.

## SYSTEM DESIGN

### Definition

Design is the first step into the development phase for any engineered product or system.

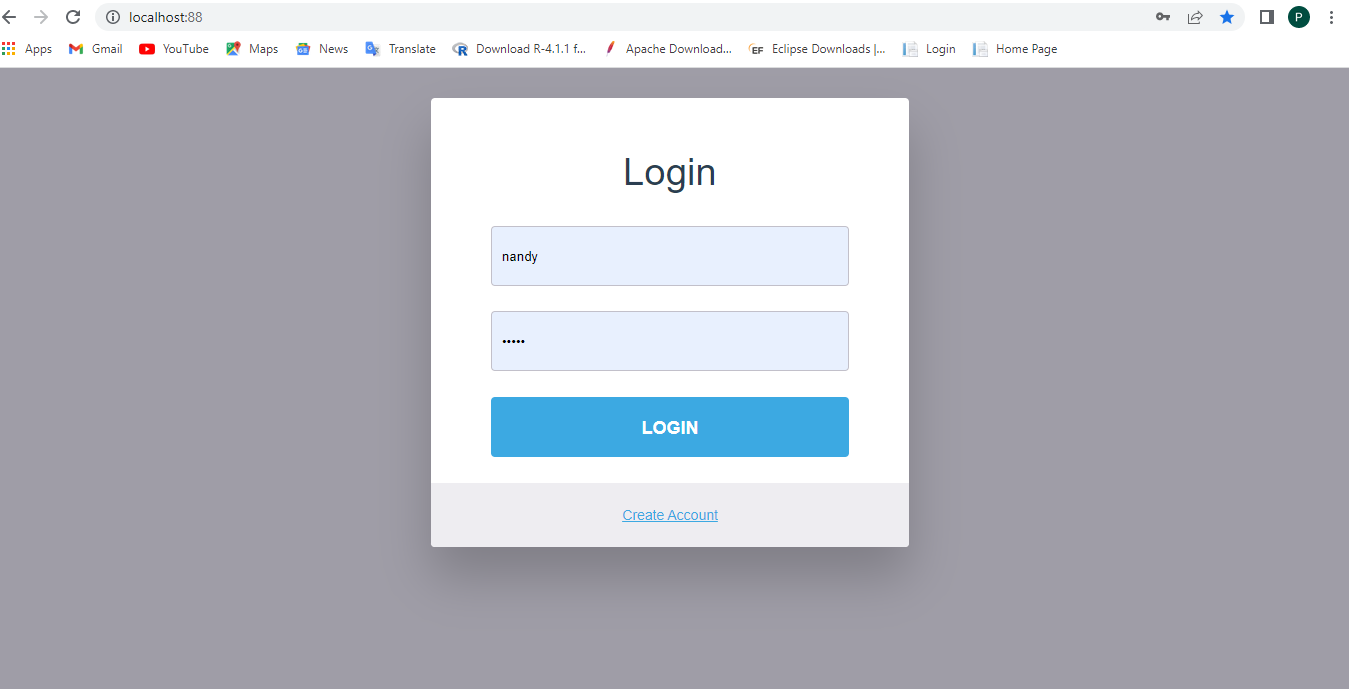
## Input Design

The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and case of use with retaining the privacy.

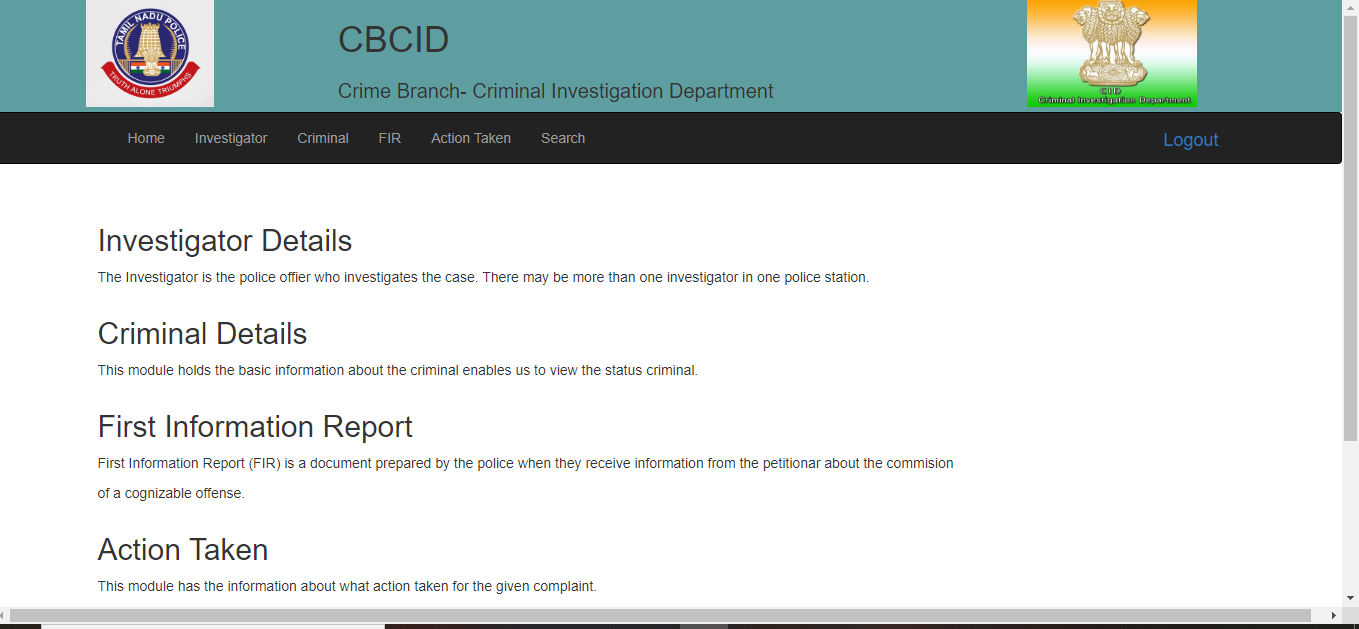
## Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. The output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively.

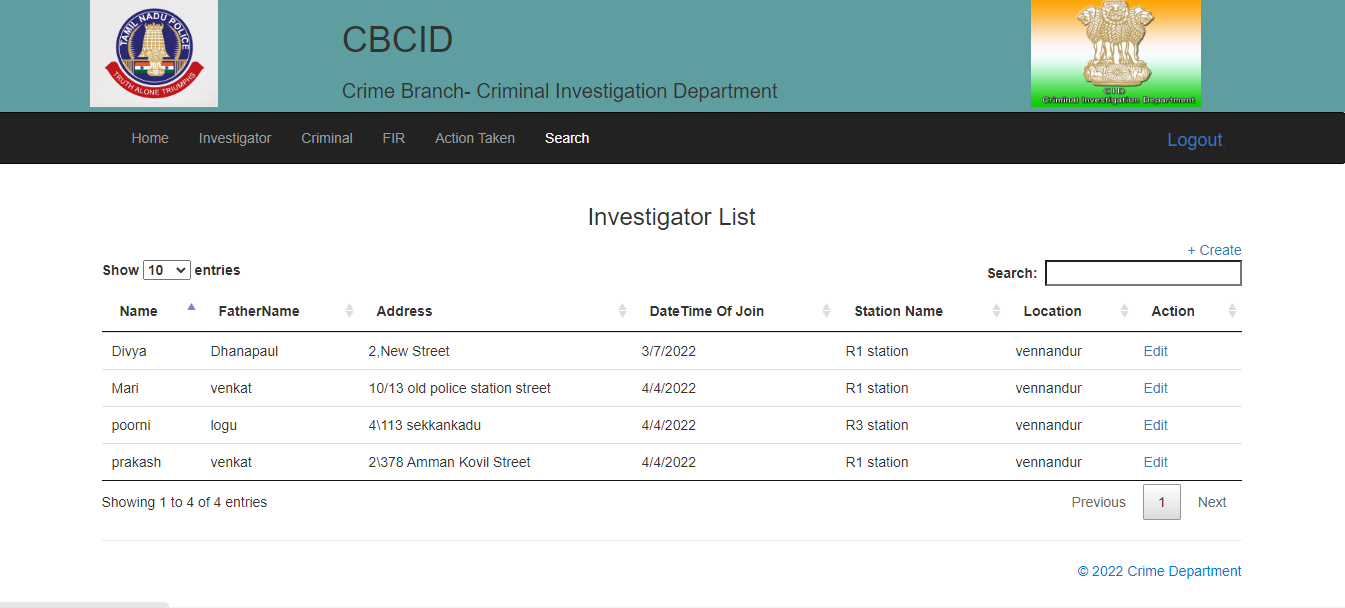
## LOG IN



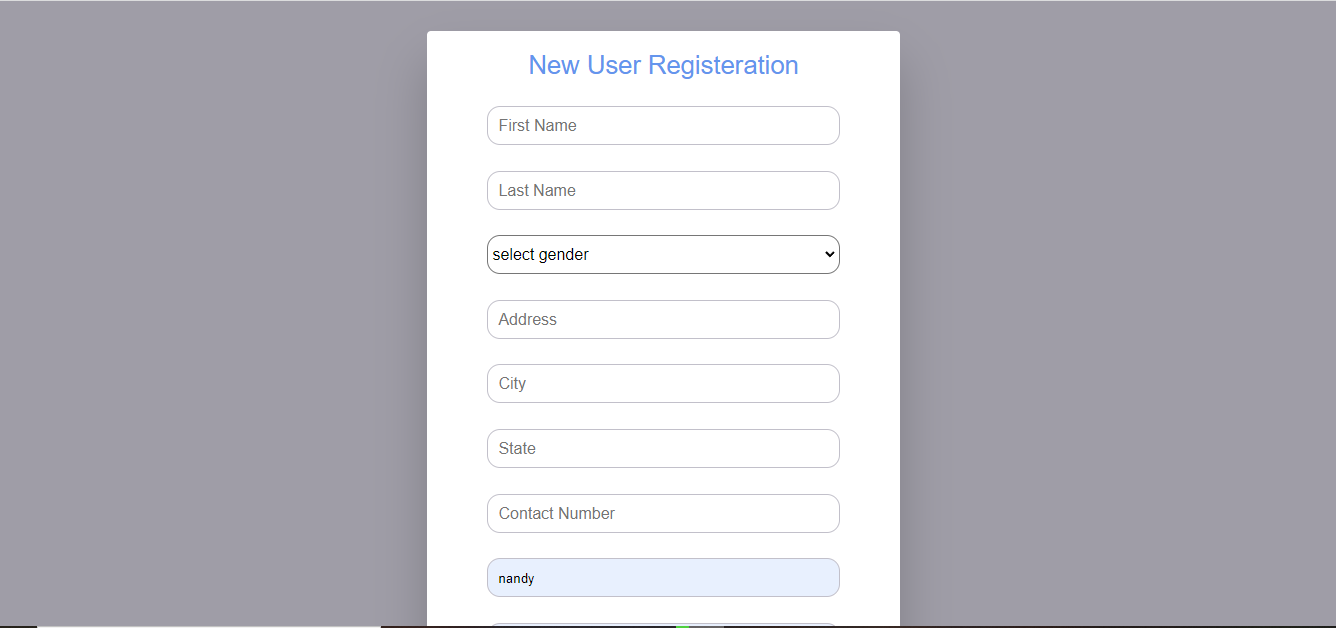
**HOME PAGE**



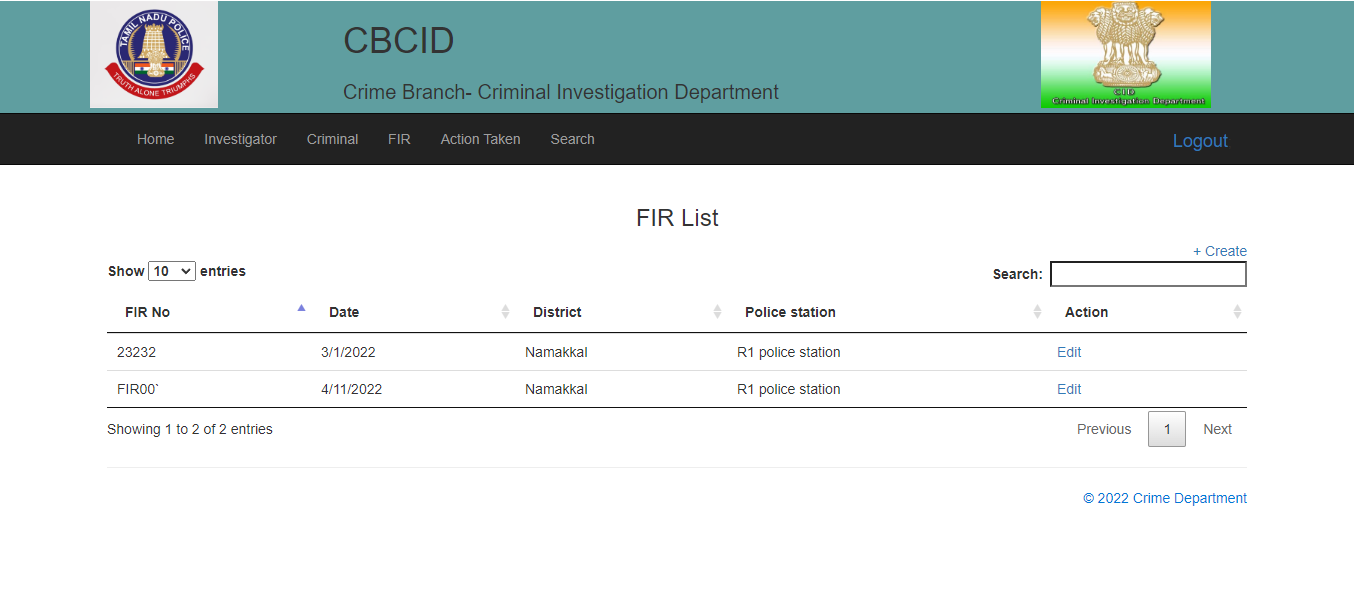
## INVESTIGATOR

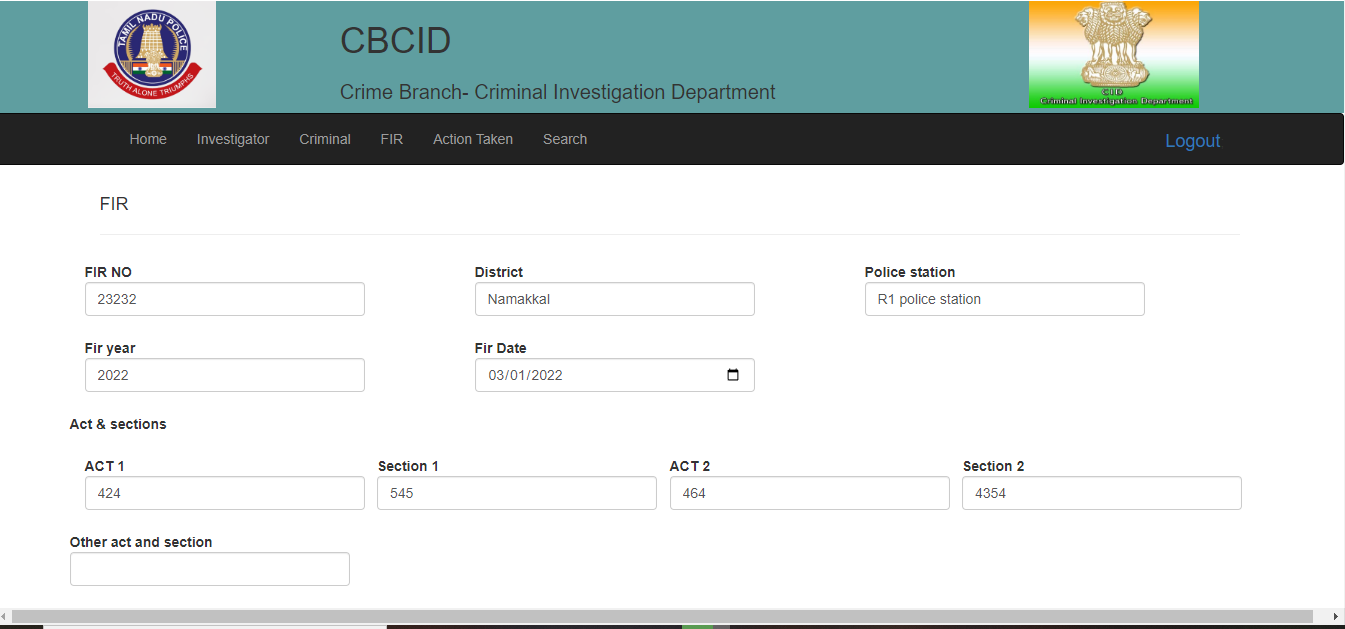


**NEW USER REGISTRATION**

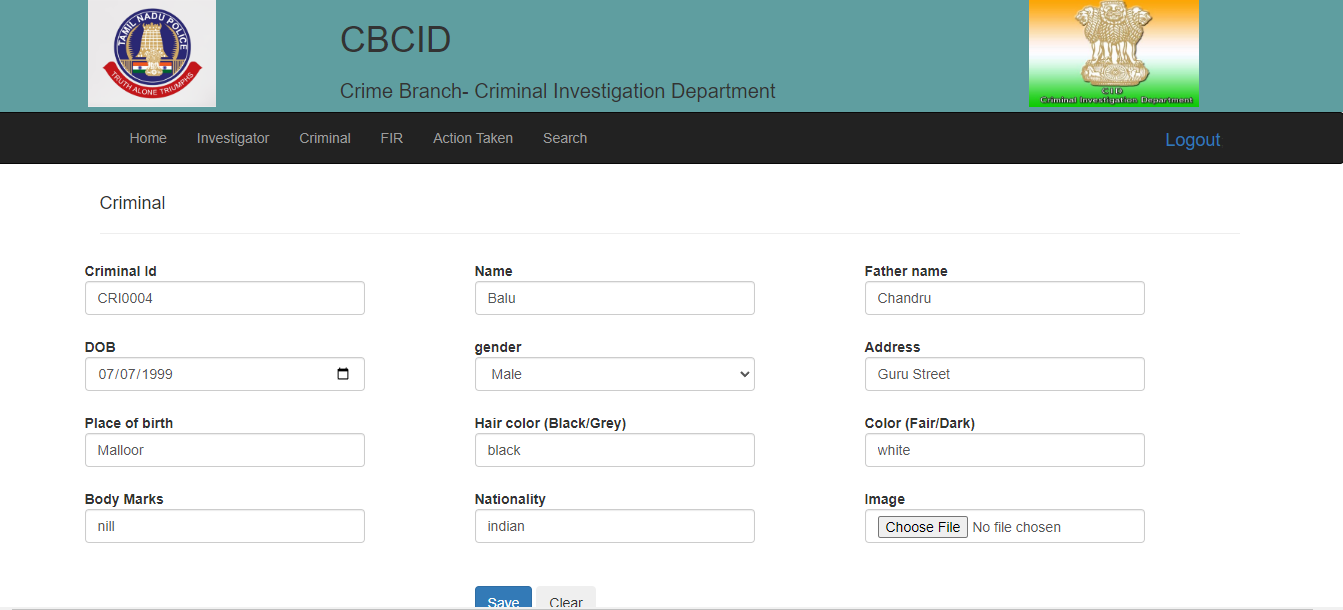


## FIRST INFORMATION REPORT

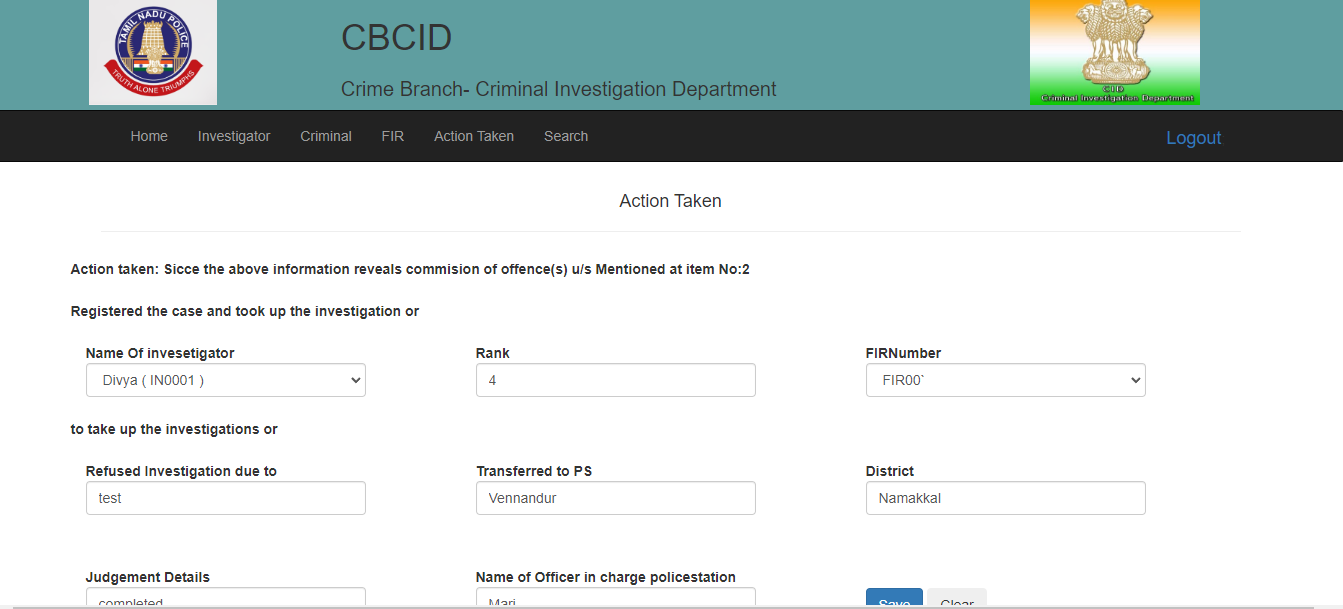




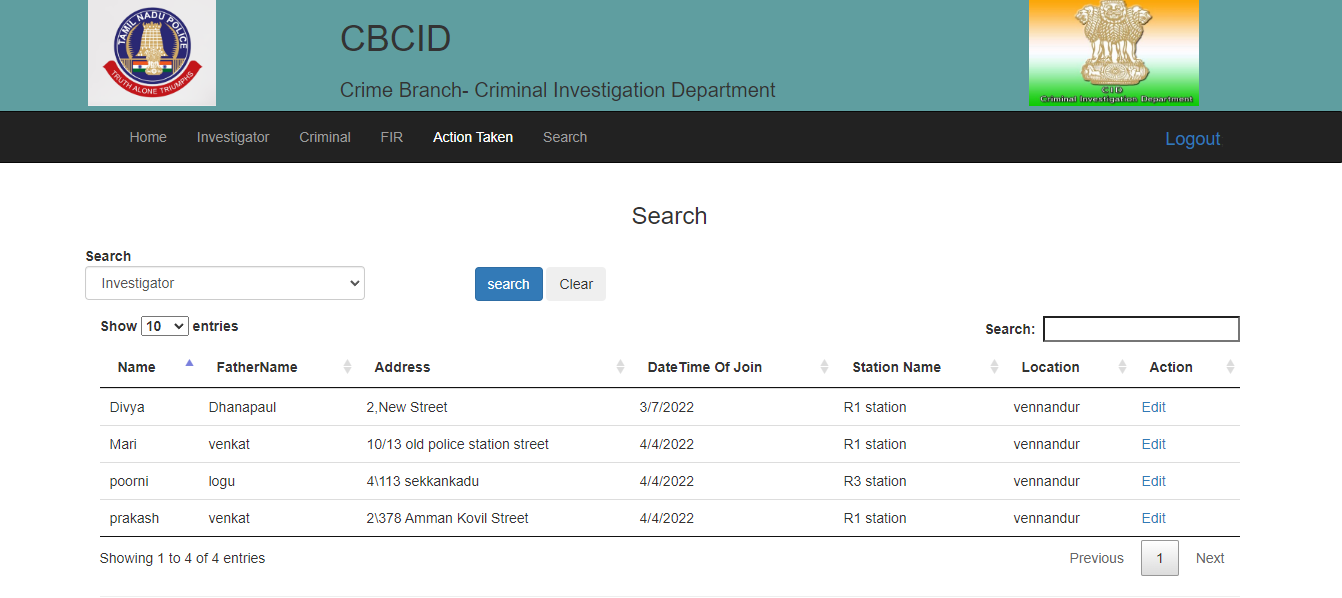
**CRIMINAL DETAILS**



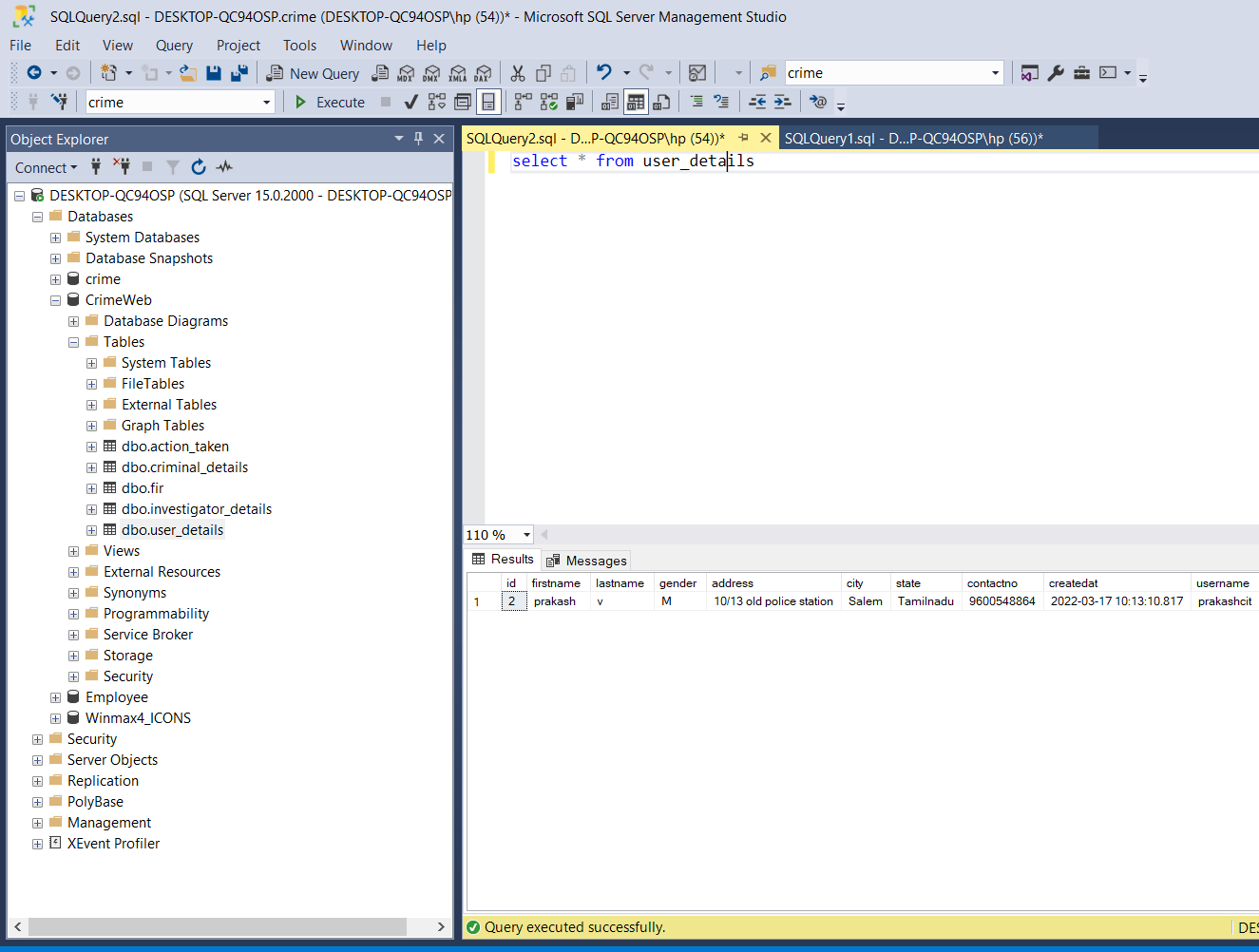
**ACTION TAKEN**



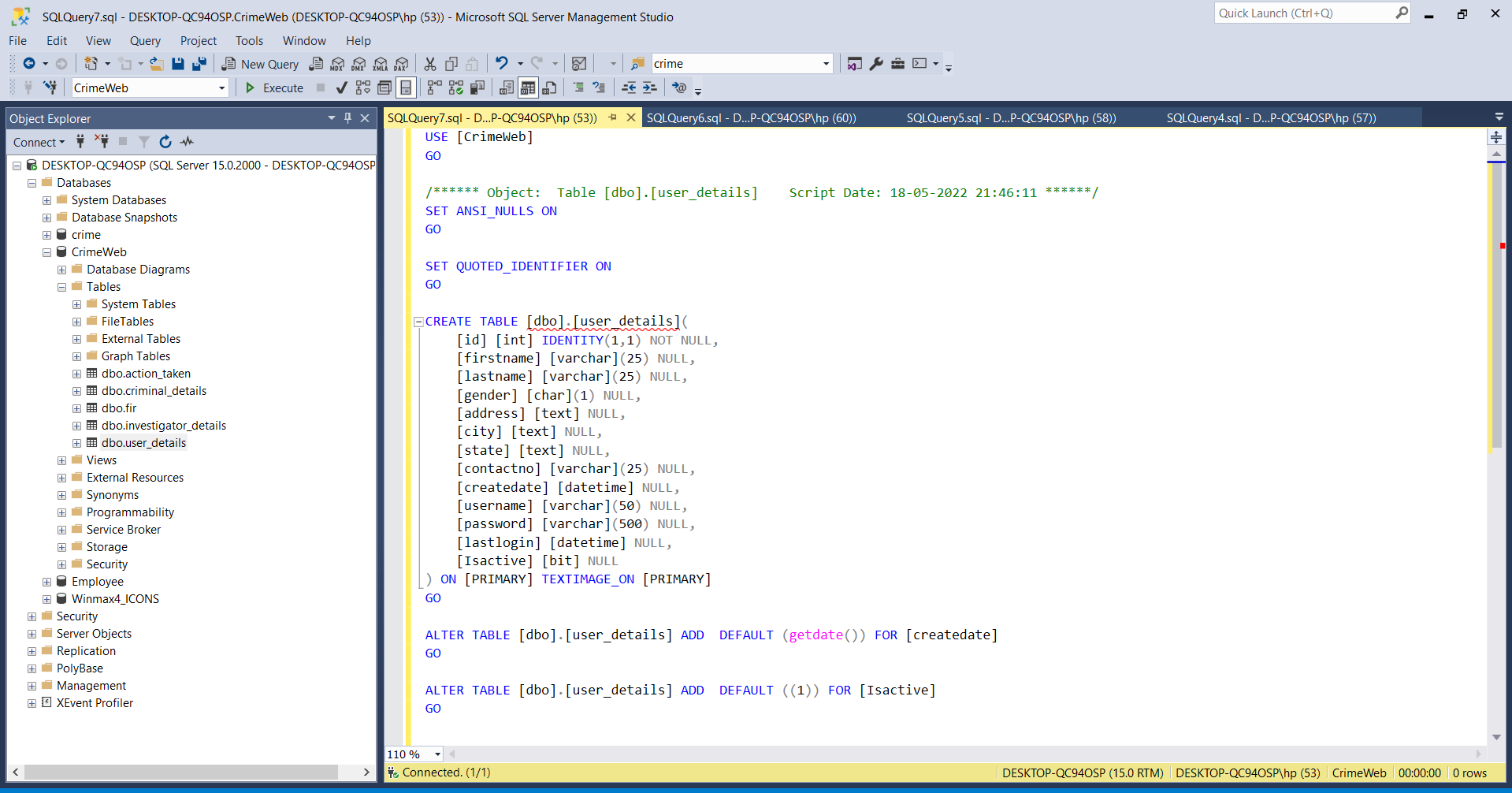
**SEARCH**



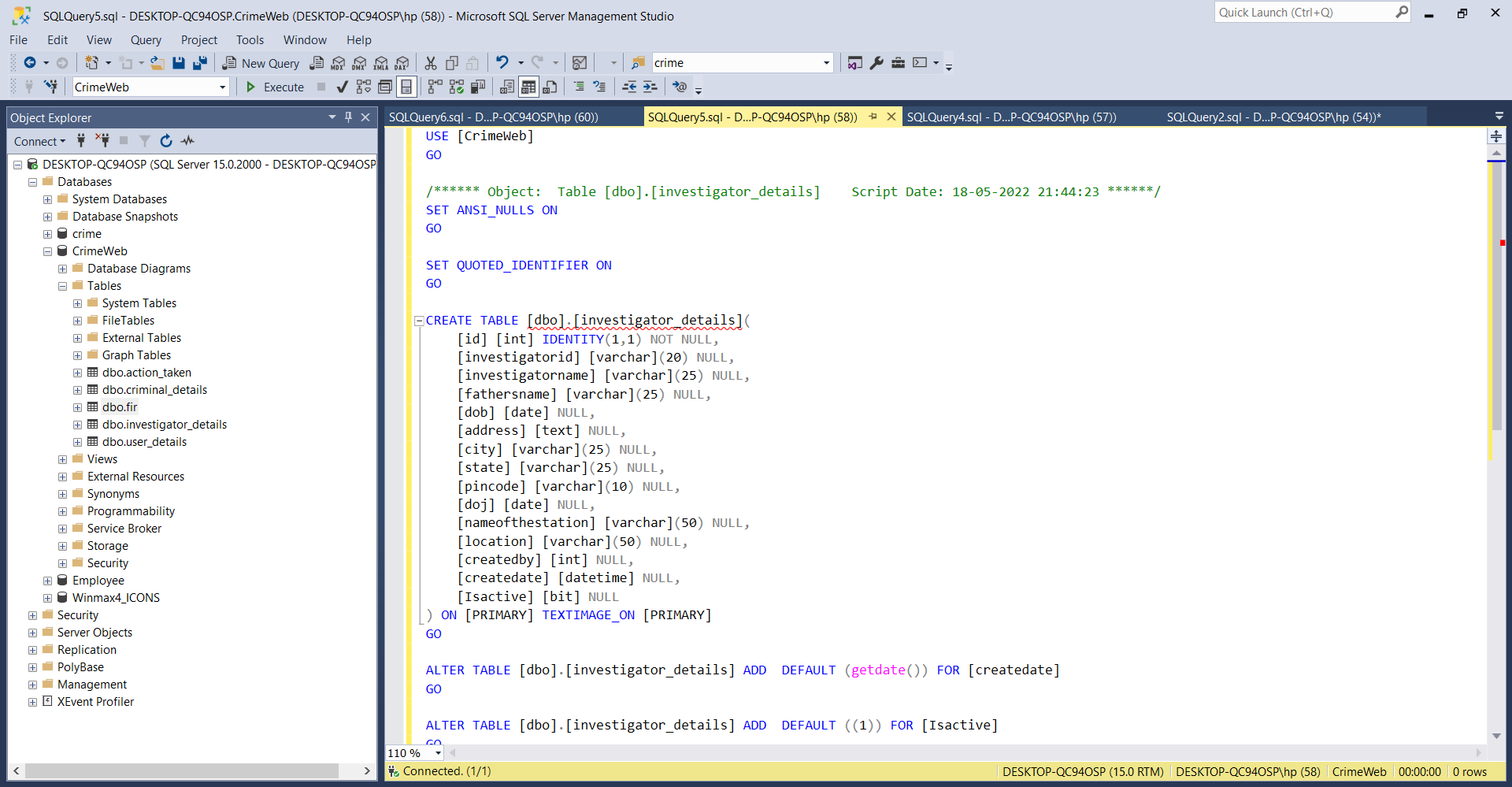
* 1. **Database Design Tables List**



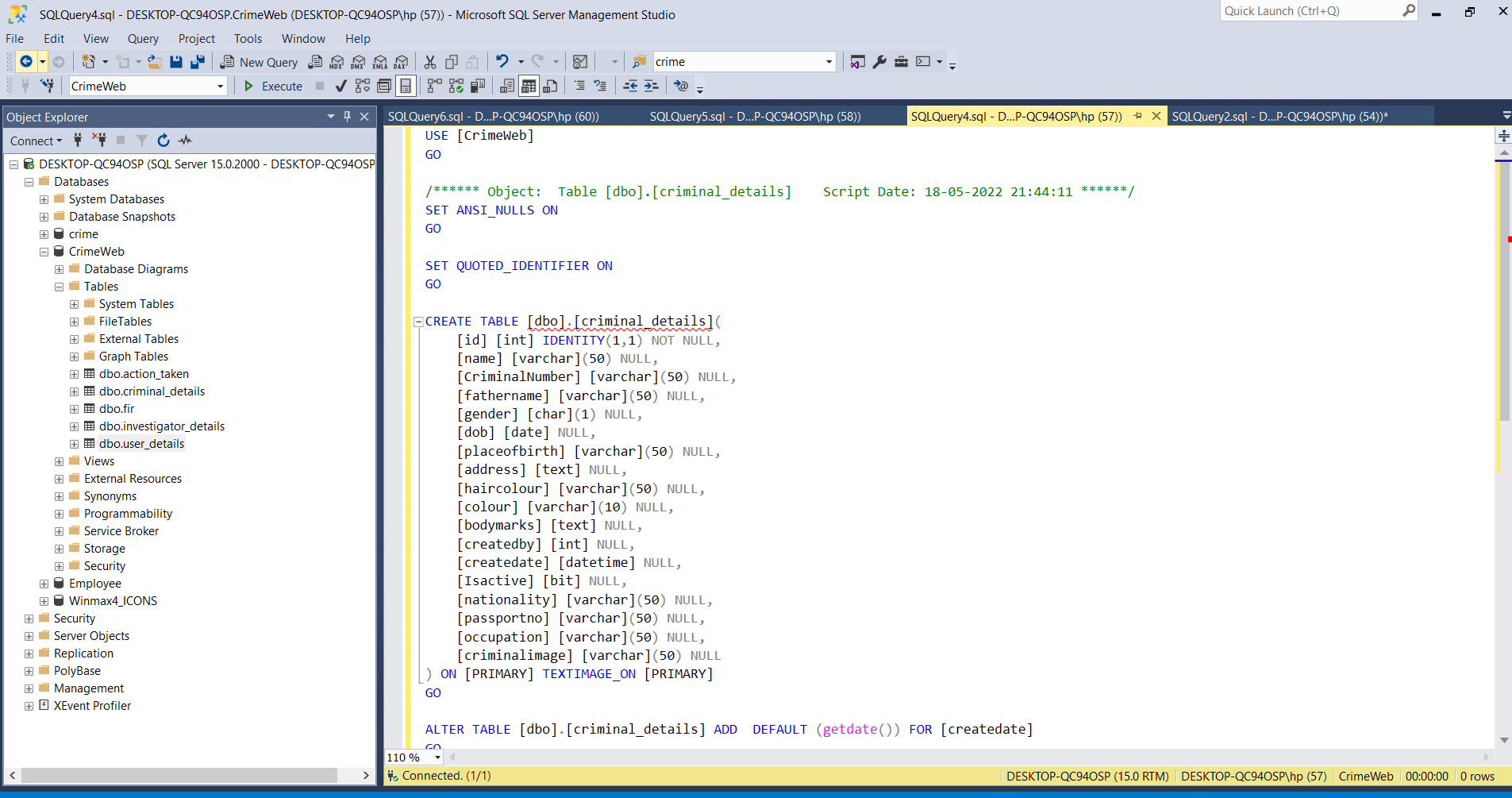
## Table – User Details



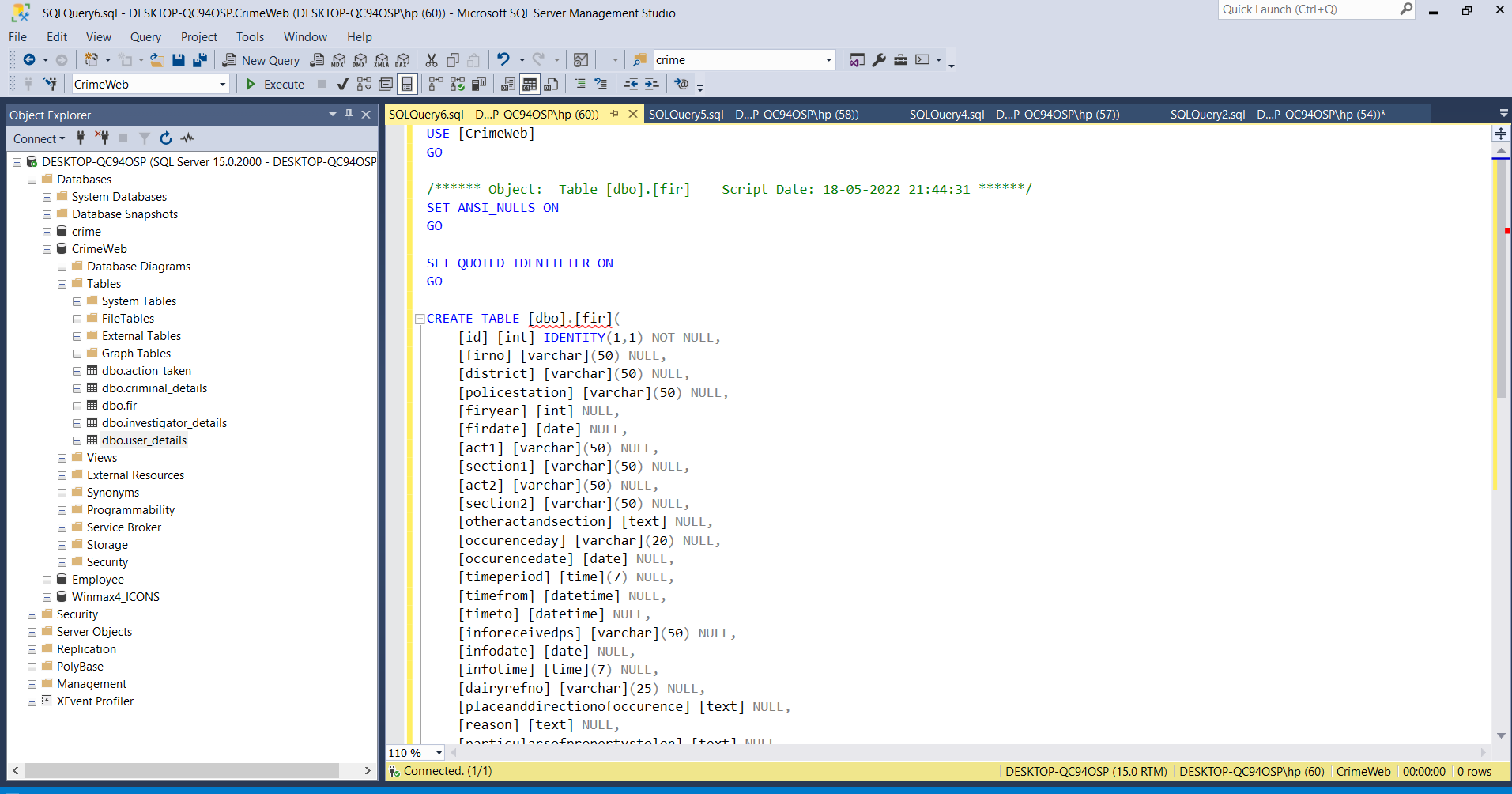
**Table – Investigator Details**



## Table – Criminal Details



**Table – FIR**



* 1. **Code Design**

FIR Model:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations;

namespace CrimeWeb.Models

{

public class FIRmodel:Basemodel

{

public int id { get; set; }

[Display(Name ="FIR NO")]

[Required(ErrorMessage ="Please enter FIR No")]

public string firno { get; set; }

[Display(Name = "District")]

[Required(ErrorMessage = "District")]

public string district { get; set; }

[Required(ErrorMessage = "Please enter Police Station")]

[Display(Name = "Police station")]

public string policestation { get; set; }

[Required(ErrorMessage = "Please enter FIR year")]

[Display(Name = "Fir year")]

public int firyear { get; set; }

[Display(Name = "Fir Date")]

[DataType(DataType.Date)]

[Required(ErrorMessage = "Please enter FIR date")]

public DateTime firdate { get; set; }

[Required(ErrorMessage = "Please enter ACT 1")]

[Display(Name = "ACT 1")]

public string act1 { get; set; }

[Required(ErrorMessage = "Please enter Section 1")]

[Display(Name = "Section 1")]

public string section1 { get; set; }

[Display(Name = "ACT 2")]

public string act2 { get; set; }

[Display(Name = "Section 2")]

public string section2 { get; set; }

[Display(Name = "Other act and section")]

public string otheractandsection { get; set; }

[Required(ErrorMessage = "Please enter Day")]

[Display(Name = "Occurence day")]

public string occurenceday { get; set; }

[Required(ErrorMessage = "Please enter Date")]

[DataType(DataType.Date)]

[Display(Name = "Occurence date")]

public DateTime occurencedate { get; set; }

[Display(Name = "Time period")]

public string timeperiod { get; set; }

[Required(ErrorMessage = "Please enter Time From")]

[Display(Name = "Time From")]

[DataType(DataType.Time)]

public TimeSpan timefrom { get; set; }

[Display(Name = "Time To")]

[Required(ErrorMessage = "Please enter Time To")]

[DataType(DataType.Time)]

public TimeSpan timeto { get; set; }

[Required(ErrorMessage = "Please enter Police station name")]

[Display(Name = "Information received at PS")]

public string inforeceivedps { get; set; }

[Display(Name = "Information date")]

[DataType(DataType.Date)]

[Required(ErrorMessage = "Please enter Information date")]

public DateTime infodate { get; set; }

[DataType(DataType.Time)]

[Display(Name = "Information Time")]

[Required(ErrorMessage = "Please enter Information Time")]

public TimeSpan infotime { get; set; }

[Required(ErrorMessage = "Please enter Ref No")]

[Display(Name = "Diary Ref No")]

public string dairyrefno { get; set; }

[Display(Name = "Place and Direction")]

[Required(ErrorMessage = "Please enter Place")]

public string placeanddirectionofoccurence { get; set; }

[Display(Name = "Reason for delay in reporting by the commision")]

public string reason { get; set; }

[Display(Name = "Particular of Property Stolen")]

[Required(ErrorMessage = "Please enter property stolen")]

public string particularsofpropertystolen { get; set; }

[Display(Name = "Total value of Property Stolen")]

[Required(ErrorMessage = "Please enter Total value stolen")]

public decimal totalvalue { get; set; }

public string fircriminals { get; set; }

[Display(Name = "FIR Criminals")]

public int[] CriminalId { get; set; }

public string FIRdatestr { get; set; }

public string Occurancedatestr { get; set; }

public string informdatestr { get; set; }

}

}

FIR Controller:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using CrimeWeb.Models;

using System.Data;

using System.Data.SqlClient;

using CrimeWeb.DataAccess;

using System.IO;

namespace CrimeWeb.Controllers

{

public class FIRController : Controller

{

private ADOHelper \_helper = new ADOHelper();

private List<DropdownModel> Lstcriminal = new List<DropdownModel>();

// GET: FIR

/// <summary>

/// All Fir details get

/// </summary>

/// <returns></returns>

public ActionResult FIRList()

{

try

{

List<FIRmodel> list = new List<FIRmodel>();

DataTable dt = \_helper.GetQuerydetails(Consvalues.FIRdetailget);

if ((dt != null) && (dt.Rows.Count > 0))

list = TabletoList.ConvertDataTable<FIRmodel>(dt);

return View(list);

}

catch (Exception ex)

{

throw ex;

}

}

[HttpGet]

public ActionResult Edit(int id)

{

try

{

FIRmodel model = new FIRmodel();

CriminalDropdown();

if (id > 0)

{

model = Editdetails(id);

model.CriminalId = model.fircriminals.Split(',').Select(int.Parse).ToArray();

model.FIRdatestr = model.firdate.ToString("yyyy-MM-dd");

model.informdatestr = model.infodate.ToString("yyyy-MM-dd");

model.Occurancedatestr = model.occurencedate.ToString("yyyy-MM-dd");

model.fircriminals = String.Join(",", Lstcriminal.Where(x => model.CriminalId.Contains(x.Id)).Select(x => x.Name).ToArray());

}

return View(model);

}

catch (Exception ex)

{

throw ex;

}

}

public ActionResult Edit(FIRmodel model)

{

try

{

if (ModelState.IsValid)

{

//double timebetween = (model.timeto - model.timefrom).TotalHours;

CriminalDropdown();

List<SqlParameter> sp = new List<SqlParameter>();

sp.Add(new SqlParameter("@Id", model.id));

sp.Add(new SqlParameter("@FirNo ", model.firno));

sp.Add(new SqlParameter("@District ", model.district));

sp.Add(new SqlParameter("@Policestation ", model.policestation));

sp.Add(new SqlParameter("@FirYear ", model.firyear));

sp.Add(new SqlParameter("@FirDate ",model.firdate));

sp.Add(new SqlParameter("@Act1 ", model.act1));

sp.Add(new SqlParameter("@Section1 ", model.section1));

sp.Add(new SqlParameter("@Act2 ", model.act2));

sp.Add(new SqlParameter("@Section2 ", model.section2));

sp.Add(new SqlParameter("@Otheractandsection ", model.otheractandsection));

sp.Add(new SqlParameter("@Occurenceday ", model.occurenceday));

sp.Add(new SqlParameter("@Occurencedate ", model.occurencedate));

sp.Add(new SqlParameter("@Timeperiod ", model.timeperiod));

sp.Add(new SqlParameter("@Timefrom ", model.timefrom));

sp.Add(new SqlParameter("@Timeto ", model.timeto));

sp.Add(new SqlParameter("@Inforeceivedps ", model.inforeceivedps));

sp.Add(new SqlParameter("@Infodate ", model.infodate));

sp.Add(new SqlParameter("@Infotime ", model.infotime));

sp.Add(new SqlParameter("@Dairyrefno ", model.dairyrefno));

sp.Add(new SqlParameter("@Placeanddirectionofoccurence ", model.placeanddirectionofoccurence));

sp.Add(new SqlParameter("@Reason ", model.reason));

sp.Add(new SqlParameter("@Particularsofpropertystolen ", model.particularsofpropertystolen));

sp.Add(new SqlParameter("@Totalvalue ", model.totalvalue));

model.fircriminals = string.Join(",", model.CriminalId);

sp.Add(new SqlParameter("@FirCriminals ", model.fircriminals));

string query = string.Format(Consvalues.FIRcheckinsert, model.firno);

DataTable dt = \_helper.GetQuerydetails(query);

if (model.id > 0)

{

FIRmodel editdetailget = new FIRmodel();

editdetailget = Editdetails(model.id);

if (editdetailget != null)

{

if (editdetailget.firno.Trim() == model.firno.Trim())

{

int NewId = \_helper.OutputResultID(Consvalues.AddFIR.ToString(), sp);

TempData["Sucessmessage"] = "Updated Sucessfully";

return RedirectToAction("FIRList");

}

else

{

ModelState.AddModelError("", "Please check FIR Number is Wrong");

}

}

}

else

{

if ((dt != null) && (dt.Rows.Count == 0))

{

int NewId = \_helper.OutputResultID(Consvalues.AddFIR.ToString(), sp);

if (NewId > 0)

{

TempData["Sucessmessage"] = "Inserted Sucessfully";

return RedirectToAction("FIRList");

}

else

{

ModelState.AddModelError("", "Please enter the all details");

}

}

else

{

ModelState.AddModelError("", "FIR Number Exist");

}

}

}

return View(model);

}

catch (Exception ex)

{

throw ex;

}

}

/// <summary>

/// Edit time values get

/// </summary>

/// <param name="id"></param>

/// <returns></returns>

private FIRmodel Editdetails(int id)

{

FIRmodel model = new FIRmodel();

string editdetails = string.Format(Consvalues.SingleFIRget, id);

DataTable dt = \_helper.GetQuerydetails(editdetails);

if ((dt != null) && (dt.Rows.Count > 0))

model = TabletoList.ConvertDataTablemodel<FIRmodel>(dt);

return model;

}

/// <summary>

/// Criminal detial load to criminal dropdown

/// </summary>

private void CriminalDropdown()

{

DataTable dt = \_helper.GetQuerydetails(Consvalues.CriminalDropdownvalue);

if ((dt != null) && (dt.Rows.Count > 0))

Lstcriminal = TabletoList.ConvertDataTable<DropdownModel>(dt);

ViewBag.Criminallist = Lstcriminal;

}

}

}

FIR View:

@model CrimeWeb.Models.FIRmodel

@using CrimeWeb.Models

@{

ViewBag.Title = "Edit";

}

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>FIR</h4>

<hr />

@Html.HiddenFor(m => m.id)

@Html.HiddenFor(m => m.createdby)

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.firno, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.firno, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.firno, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.district, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.district, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.district, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.policestation, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.policestation, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.policestation, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.firyear, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.firyear, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.firyear, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.firdate, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.firdate, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.firdate, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="form-group">

@Html.Label("Act & sections", htmlAttributes: new { @class = "control-label" })

</div>

<div class="col-md-3">

<div class="form-group">

@Html.LabelFor(model => model.act1, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.act1, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.act1, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-3">

<div class="form-group">

@Html.LabelFor(model => model.section1, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.section1, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.section1, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-3">

<div class="form-group">

@Html.LabelFor(model => model.act2, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.act2, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.act2, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-3">

<div class="form-group">

@Html.LabelFor(model => model.section2, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.section2, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.section2, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="form-group">

@Html.LabelFor(model => model.otheractandsection, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.otheractandsection, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.otheractandsection, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="row">

<div class="form-group">

@Html.Label("Occurrence Of Offence", htmlAttributes: new { @class = "control-label" })

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Day", htmlAttributes: new { @class = "control-label" })

<div>

@Html.DropDownList("occurenceday", new List<SelectListItem>()

{

new SelectListItem{Text="select",Value=""},

new SelectListItem{Text="Sunday",Value="Sunday"},

new SelectListItem{Text="Monday",Value="Monday"},

new SelectListItem{Text="Tuesday",Value="Tuesday"},

new SelectListItem{Text="Wednesday",Value="Wednesday"},

new SelectListItem{Text="Thursday",Value="Thursday"},

new SelectListItem{Text="Friday",Value="Friday"},

new SelectListItem{Text="Saturday",Value="Saturday"},

}, new { @class = "form-control" })

<span id="errorday" class="text-danger" style="display: none; width: 172px;">Please select the Day</span>

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Date", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.occurencedate, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.occurencedate, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.Label("Time Period", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.timeperiod, new { htmlAttributes = new { @class = "form-control",@readonly="readonly" } })

@Html.ValidationMessageFor(model => model.timeperiod, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Time From", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.timefrom, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.timefrom, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Time To", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.timeto, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.timeto, "", new { @class = "text-danger" })

<span id="errortime" class="text-danger" style="display: none; width: 172px;">start time should be smaller than end time!</span>

</div>

</div>

</div>

</div>

<div class="row">

<div class="form-group">

@Html.Label("Information Received At", htmlAttributes: new { @class = "control-label" })

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Police station", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.inforeceivedps, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.inforeceivedps, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Date", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.infodate, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.infodate, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Time", htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.infotime, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.infotime, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="form-group">

@Html.Label("Other Information", htmlAttributes: new { @class = "control-label" })

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.dairyrefno, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.dairyrefno, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.dairyrefno, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.placeanddirectionofoccurence, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.placeanddirectionofoccurence, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.placeanddirectionofoccurence, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.reason, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.reason, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.reason, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.particularsofpropertystolen, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.particularsofpropertystolen, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.particularsofpropertystolen, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.totalvalue, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.totalvalue, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.totalvalue, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.Label("Criminal", htmlAttributes: new { @class = "control-label" })

<div style="margin-top: 10px">

@Html.DropDownListFor(model=>model.CriminalId, new SelectList(ViewBag.Criminallist as List<DropdownModel>, "Id", "Name", null), new { @multiple = "multiple", @class = "form-control crimimalmulti" })

<span id="errorcriminal" class="text-danger" style="display: none; width: 172px;">Please select the criminal</span>

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4"></div>

<div class="col-md-4"></div>

<div class="col-md-4">

<div class="form-group">

<div style="margin-top: 28px;">

<input type="submit" id="btnfircreate" value="Save" class="btn btn-primary" />

<input type="reset" value="Clear" class="btn btn-secondary" />

</div>

</div>

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to List", "FIRList")

</div>

<script type="text/javascript" src="~/Scripts/Commonjs/jquery-2.2.4.min.js"></script>

<script src="~/Scripts/Commonjs/jquery.multi-select.js"></script>

<script>var $jq1111 = jQuery.noConflict(true);</script>

<script>

$jq1111(document).ready(function () {

var modeldata = @Html.Raw(Json.Encode(Model));

if (modeldata.id > 0) {

$('#firdate').val(modeldata.FIRdatestr)

$('#occurencedate').val(modeldata.Occurancedatestr)

$('#infodate').val(modeldata.informdatestr)

$('.multi-select-button').text(modeldata.fircriminals);

$.each(modeldata.CriminalId, function (item,value) {

// $("input[value='"+value+"']").prop('checked', true);

$(".crimimalmulti option[value='" + value +"']").prop("selected", true)

})

}

$jq1111('.crimimalmulti').multiSelect();

});

</script>

<script type="text/javascript">

$(document).ready(function () {

@\*var modeldata = @Html.Raw(Json.Encode(Model));

if (modeldata.id > 0) {

$('#firdate').val(modeldata.FIRdatestr)

$('#occurencedate').val(modeldata.Occurancedatestr)

$('#infodate').val(modeldata.informdatestr)

$('.multi-select-button').text(modeldata.fircriminals);

$.each(modeldata.CriminalId, function (item,value) {

$("input[value='"+value+"']").prop('checked', true);

})

}\*@

$('#btnfircreate').click(function(){

if ($('.crimimalmulti').val() == null) {

$('#errorcriminal').css("display", "block")

return false;

} else {

$('#errorcriminal').css("display", "none")

}

if ( $('#occurenceday').val()=="") {

$('#errorday').css("display", "block")

return false;

} else {

$('#errorday').css("display", "none")

}

var timefrom = new Date();

temp = $('#timefrom').val().split(":");

timefrom.setHours((parseInt(temp[0]) - 1 + 24) % 24);

timefrom.setMinutes(parseInt(temp[1]));

var timeto = new Date();

temp = $('#timeto').val().split(":");

timeto.setHours((parseInt(temp[0]) - 1 + 24) % 24);

timeto.setMinutes(parseInt(temp[1]));

if (timeto < timefrom) {

$('#errortime').css("display", "block")

return false;

} else {

var diff = timeto - timefrom;

var diffSeconds = diff / 1000;

var HH = Math.floor(diffSeconds / 3600);

var MM = Math.floor(diffSeconds % 3600) / 60;

var formatted = ((HH < 10) ? ("0" + HH) : HH) + ":" + ((MM < 10) ? ("0" + MM) : MM)

$('#timeperiod').val(formatted);

$('#errortime').css("display", "none")

}

})

})

</script>

Investigator Model:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.ComponentModel.DataAnnotations;

namespace CrimeWeb.Models

{

public class InvestigatorModel:Basemodel

{

public int id { get; set; }

[Display(Name = "Investigator Id")]

[Required(ErrorMessage = "Please enter Investigator Id")]

public string investigatorid { get; set; }

[Required(ErrorMessage ="Please enter the Name"),MaxLength(25)]

[Display(Name = "Investigator Name")]

public string investigatorname { get; set; }

[MaxLength(25)]

[Display(Name = "Father Name")]

public string fathersname { get; set; }

[Required(ErrorMessage = "Please enter DOB"), DataType(DataType.Date)]

[Display(Name = "DOB")]

public DateTime dob { get; set; }

[Required(ErrorMessage = "Please enter the Address")]

[Display(Name = "Address")]

public string address { get; set; }

[Display(Name = "City")]

public string city { get; set; }

[Display(Name = "State")]

public string state { get; set; }

[Display(Name = "Pincode")]

[StringLength(6,ErrorMessage ="Pincode have 6 digit")]

public string pincode { get; set; }

[Required(ErrorMessage ="Please enter date of join"),DataType(DataType.Date)]

[Display(Name = "Date of Join")]

public DateTime doj { get; set; }

[Display(Name = "Station Name")]

[Required(ErrorMessage = "Please enter station name ")]

public string nameofthestation { get; set; }

[Display(Name = "Location")]

public string location { get; set; }

public string DOBstring { get; set; }

public string DOJstring { get; set; }

}

}

Investigator Controller:

using CrimeWeb.DataAccess;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using CrimeWeb.Models;

using System.Data;

using System.Data.SqlClient;

namespace CrimeWeb.Controllers

{

public class InvestigatorController : Controller

{

private ADOHelper \_helper = new ADOHelper();

// GET: Investigator

public ActionResult InvestigatorList()

{

try

{

List<InvestigatorModel> list = new List<InvestigatorModel>();

DataTable dt = \_helper.GetQuerydetails(Consvalues.Investigatordetailget);

if ((dt != null) && (dt.Rows.Count > 0))

list = TabletoList.ConvertDataTable<InvestigatorModel>(dt);

return View(list);

}

catch (Exception ex)

{

throw ex;

}

}

[HttpGet]

public ActionResult Edit(int id)

{

try

{

InvestigatorModel model = new InvestigatorModel();

if (id > 0)

{

model = Editdetails(id);

model.DOBstring = model.dob.ToString("yyyy-MM-dd");

model.DOJstring = model.doj.ToString("yyyy-MM-dd");

}

return View(model);

}

catch (Exception ex)

{

throw ex;

}

}

public ActionResult Edit(InvestigatorModel model)

{

try

{

if (ModelState.IsValid)

{

List<SqlParameter> sp = new List<SqlParameter>();

sp.Add(new SqlParameter("@Id", model.id));

sp.Add(new SqlParameter("@Investigatorid", model.investigatorid));

sp.Add(new SqlParameter("@Investigatorname", model.investigatorname));

sp.Add(new SqlParameter("@Fathersname", model.fathersname));

sp.Add(new SqlParameter("@DOB", model.dob));

sp.Add(new SqlParameter("@Address", model.address));

sp.Add(new SqlParameter("@City", model.city));

sp.Add(new SqlParameter("@State", model.state));

sp.Add(new SqlParameter("@Pincode", model.pincode));

sp.Add(new SqlParameter("@DOJ", model.doj));

sp.Add(new SqlParameter("@Nameofthestation", model.nameofthestation));

sp.Add(new SqlParameter("@Location", model.location));

sp.Add(new SqlParameter("@Createdby", model.createdby));

string query = string.Format(Consvalues.Investigatorcheckinsert, model.investigatorid);

DataTable dt = \_helper.GetQuerydetails(query);

if (model.id > 0)

{

InvestigatorModel editdetailget = new InvestigatorModel();

editdetailget = Editdetails(model.id);

if (editdetailget != null)

{

if (editdetailget.investigatorid.Trim() == model.investigatorid.Trim())

{

int NewId = \_helper.OutputResultID(Consvalues.AddInvstigater.ToString(), sp);

TempData["Sucessmessage"] = "Updated Sucessfully";

return RedirectToAction("InvestigatorList");

}

else

{

ModelState.AddModelError("", "Please check Investigator Id is Wrong");

}

}

}

else

{

if ((dt != null) && (dt.Rows.Count == 0))

{

int NewId = \_helper.OutputResultID(Consvalues.AddInvstigater.ToString(), sp);

if(NewId > 0)

{

TempData["Sucessmessage"] = "Inserted Sucessfully";

return RedirectToAction("InvestigatorList");

}

else

{

ModelState.AddModelError("", "Please enter the all details");

}

}

else

{

ModelState.AddModelError("", "Investigator Id Exist");

}

}

}

return View(model);

}

catch (Exception ex)

{

throw ex;

}

}

private InvestigatorModel Editdetails(int id)

{

InvestigatorModel model= new InvestigatorModel();

string editdetails = string.Format(Consvalues.SingleInvestigatorget, id);

DataTable dt = \_helper.GetQuerydetails(editdetails);

if ((dt != null) && (dt.Rows.Count > 0))

model = TabletoList.ConvertDataTablemodel<InvestigatorModel>(dt);

return model;

}

}

}

Investigator View:

@model CrimeWeb.Models.InvestigatorModel

@{

ViewBag.Title = "Edit";

}

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4>Investigator</h4>

<hr />

@Html.HiddenFor(m => m.id, new { Id = "investid" })

@Html.HiddenFor(m => m.createdby)

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.investigatorid, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.investigatorid, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.investigatorid, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.investigatorname, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.investigatorname, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.investigatorname, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.fathersname, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.fathersname, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.fathersname, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.dob, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.dob, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.dob, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.address, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.address, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.address, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.city, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.city, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.city, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.state, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.state, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.state, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.pincode, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.pincode, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.pincode, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.doj, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.doj, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.doj, "", new { @class = "text-danger" })

</div>

</div>

</div>

</div>

<div class="row">

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.nameofthestation, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.nameofthestation, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.nameofthestation, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

@Html.LabelFor(model => model.location, htmlAttributes: new { @class = "control-label" })

<div>

@Html.EditorFor(model => model.location, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.location, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="col-md-4">

<div class="form-group">

<div style="margin-top: 28px;">

<input type="submit" value="Save" class="btn btn-primary" />

<input type="reset" value="Clear" class="btn btn-secondary" />

</div>

</div>

</div>

</div>

</div>

}

<div style="text-align:right">

@Html.ActionLink("Back to List", "InvestigatorList")

</div>

<script>

$(document).ready(function () {

var modeldata = @Html.Raw(Json.Encode(Model));

if (modeldata.id > 0) {

$('#dob').val(modeldata.DOBstring)

$('#doj').val(modeldata.DOJstring)

}

})

</script>

# TESTING AND IMPLEMENTATION

## TESTING AND IMPLEMENTATION

### SYSTEM TESTING

System testing is the state of implementation, which is aimed at ensuring that the System Works accurate and efficient as expect before, live operation, commences. It certifies that the whole set of programs hang together system testing requires a test plan, that consist of several key activities and step for run program, string, system and user acceptance testing. the implementation of newly design package is important in adapting a successful new system.

Testing is important stage in software development system test is implementation should be as confirmation that all is correct and opportunity to show the user that the system works as they expected it accounts the largest percentage of technical effort in software development process

Testing phase development phase that validates the code against the functional specification. testing is a vital the achievement of the system goals the objective of testing is to discover errors, to fulfill this objective a series of test step such as the unit test, integration, validation and system test where planned and executed.

### UNIT TESTING

Unit testing are called module testing. The following modules have been tested

* + - Login module - valid username and password.
    - New User Registration - valid user and specification.
    - FIR module - valid FIR fields and structure
    - Search module - valid documents retrieved

### INTEGRATION TESTING

Following modules have been integrated for evaluating the efficiency of interface

* + - User Module - Admin log
    - Investigator module - insert, update, clear, back
    - FIR module - insert, update, delete, clear, back, retrieve.

The acceptance is the final stage of the user the various possibilities of the data are entered and the result are tested.

### VAILIDATION TESTING

Software validation is achieved through a series of test that demonstrates the conformity and requirement thus the proposed system under consideration has to be tested by validation and found to be working satisfactorily

### SYSTEM IMPLEMENTATION

Implementation is the final and important phase, the most critical stage in achieving a successful new system and giving the user confidence. That the new system will work be effective. The system can be implemented only after through testing is done and if it found to working according to the specification

# CONCLUSION

## CONCLUSION

**CRIME FILE MANAGEMENT SYSTEM** initiates the objective of providing the user with customized and powerful complaint registration and process management system side software. The software is built with all options such as complaints registration and search. All the requirements specified during the analysis and design phase are the fully meet, thus resulting in the formation of good software. This interface provided is very user friendly and flexible for all times.

# FUTURE ENHANCEMENTS AND BIBLIOGRAPY

## FUTURE ENHANCEMENTS

* The system can be update as online application.
* The main advantage of online application is that the person can report the crime anytime from anywhere.
* By the future technology user can view the case detail and progress of the complaints on their mobile phones
* Bio-metric recognition technology can apply. User or witness of the crime can give the physical information of the person. Search as finger print sensors, retina sensors, Iris sensors and face recognition etc.
* For example: The finger prints of the criminal/accused and investigator can be taken as the signature.
* The case file of the criminals and the crime details can even retrieve by using the fingerprint, since each finger print will be unique and so the random number generated for that particular finger print will not repeat at any cost.

## BIBLIOGRAPHY

* 1. **BOOKS**
* A complete Guide to Programming ASP.NET
* A complete ASP.NET Training Course
* Advanced Programming Using ASP.NET
* SQL Server Relational Database Design and Implementation.
  1. **SITE ADDRESS**
* [www.associatedcontent.com](http://www.associatedcontent.com)
* [www.members.tripod.com](http://www.members.tripod.com)
* [www.seminarprojects.com](http://www.seminarprojects.com)
* [www.scribd.com](http://www.scribd.com)