**Assignment: NADRA Management System GUI**



Session: 2021 – 2024

**Submitted by:**

Rayan Rasheed 2021-CS-59

**Supervised by:**

Dr. Awais Hassan

Department of Computer Science

**University of Engineering and Technology**

Table of Contents

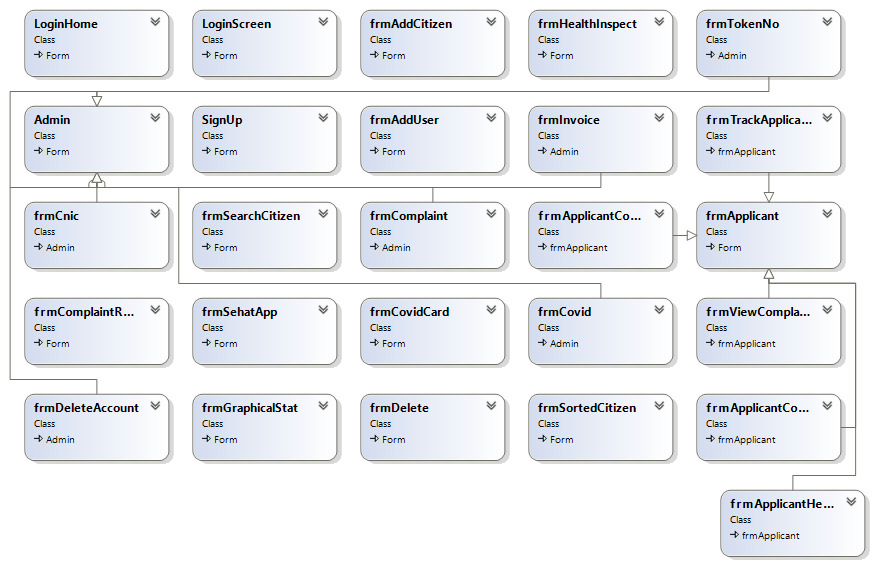
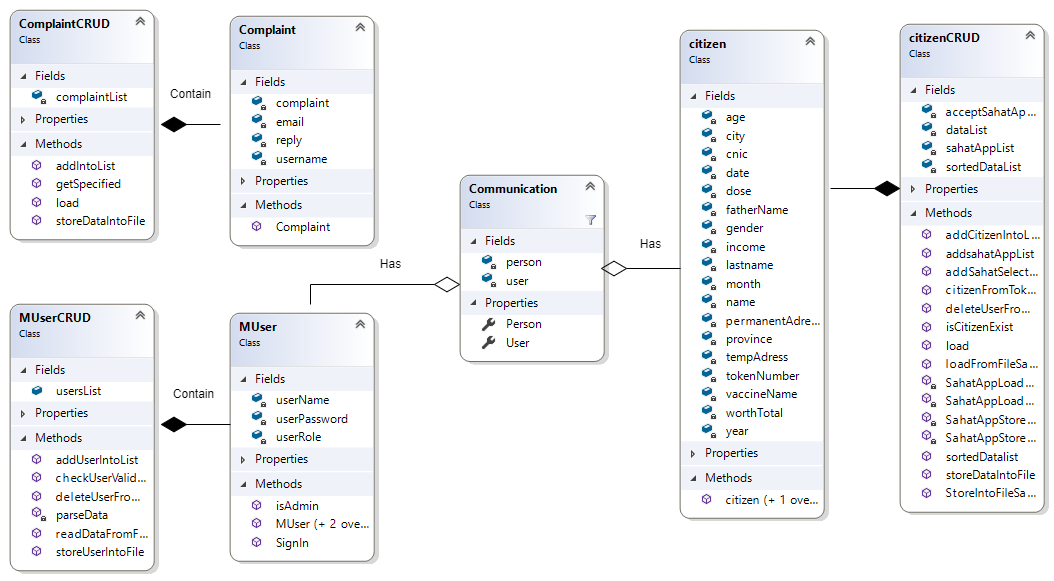
Class Diagram 3

Sequence Diagram4

Wireframes 7

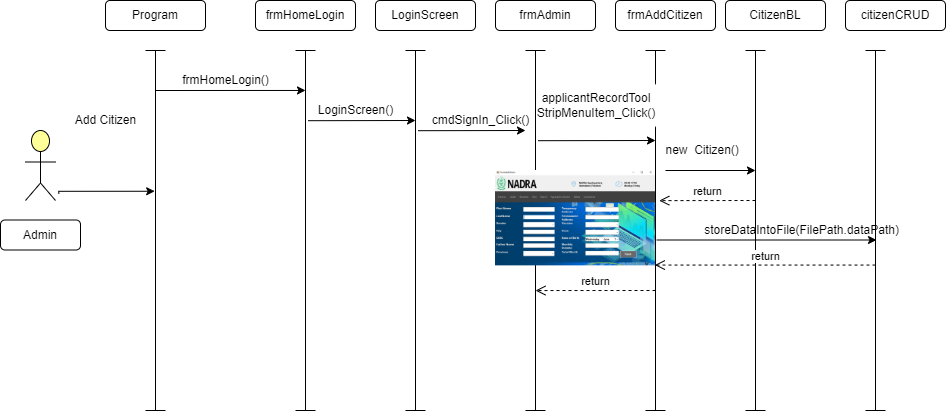
Full Code17

**Step 1: Class Diagrams**

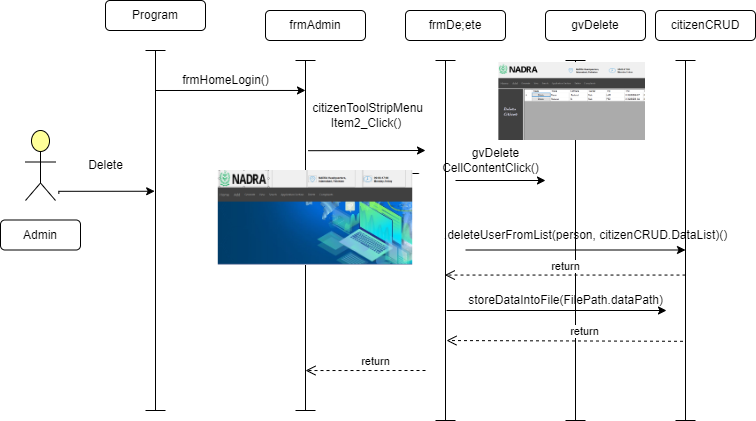


**Step 2: Sequence Diagram (CRUD)**

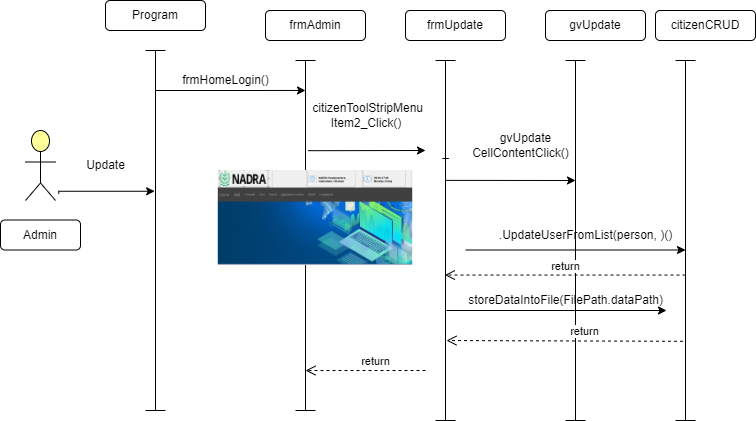
**Create**

****

**Delete:**

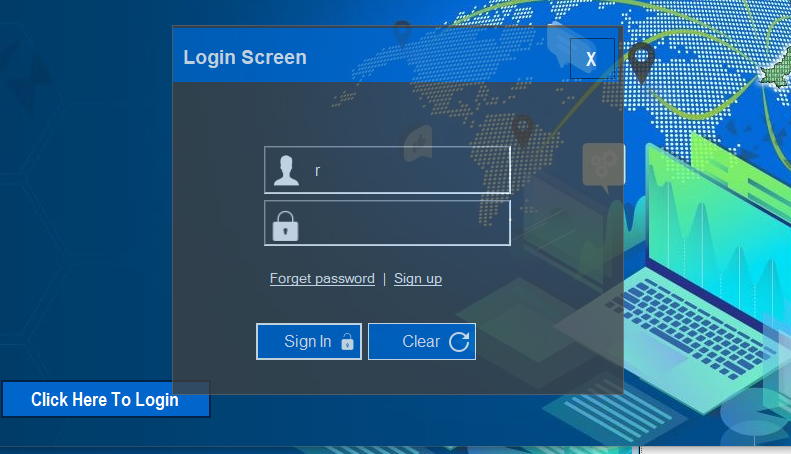


**Update:**

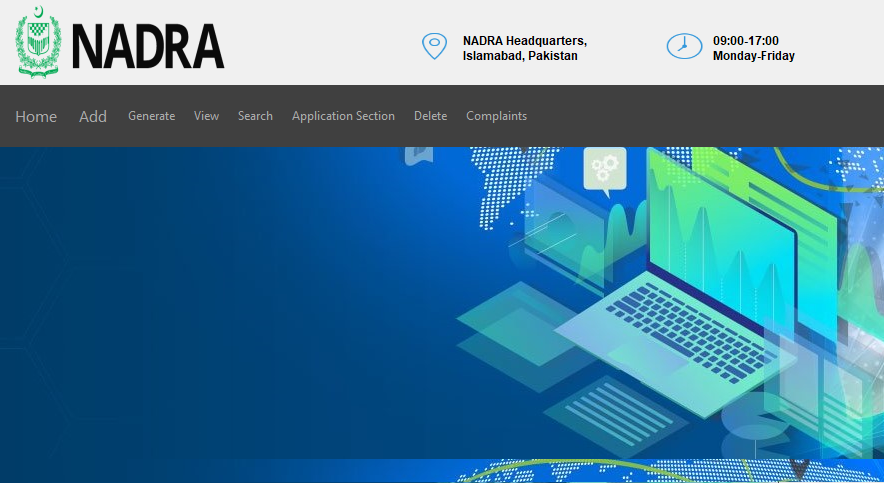
****

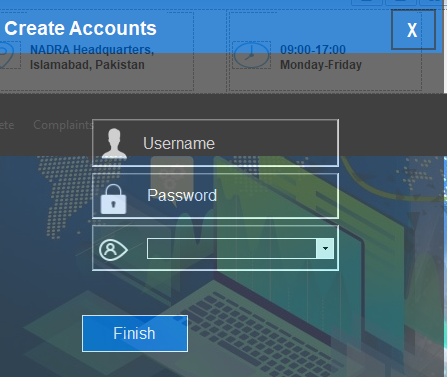
**Retreive:**

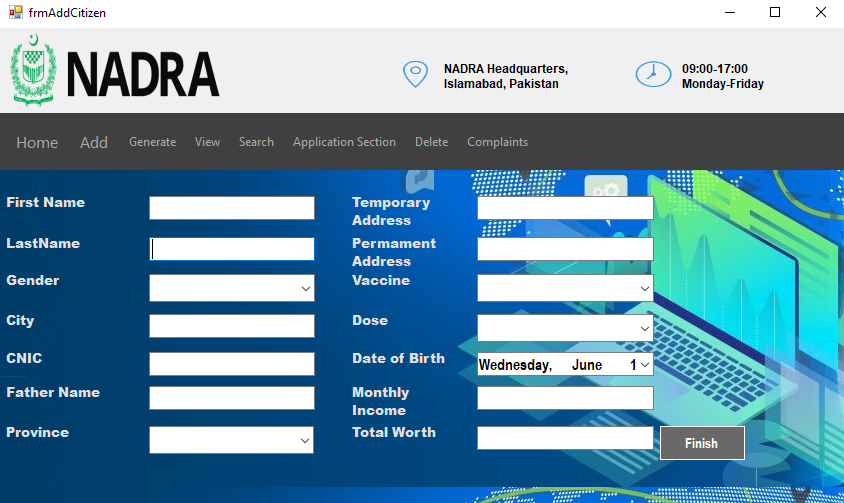


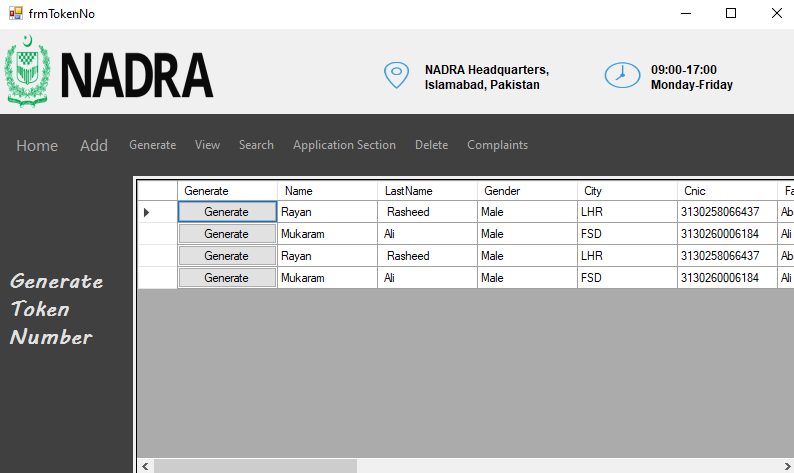
**Wireframes:**

**Step 4: Domain Model with Multiplicity**

****

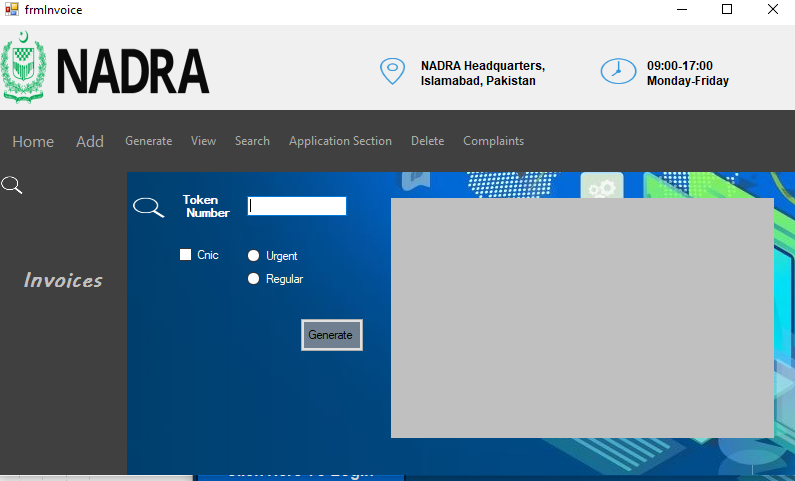
****

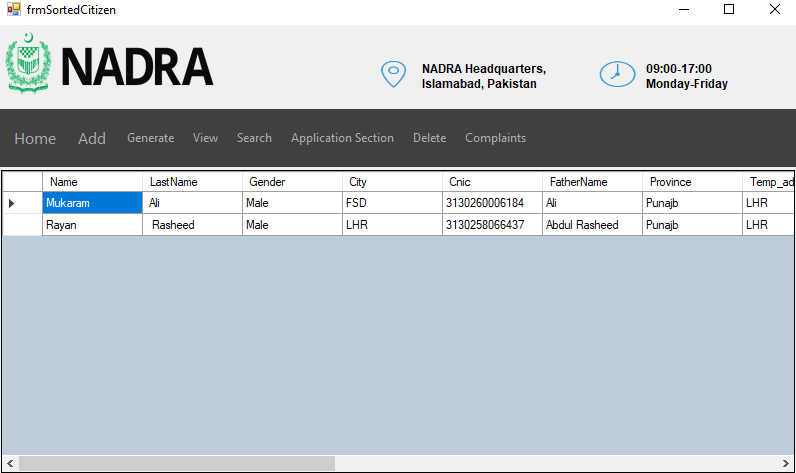


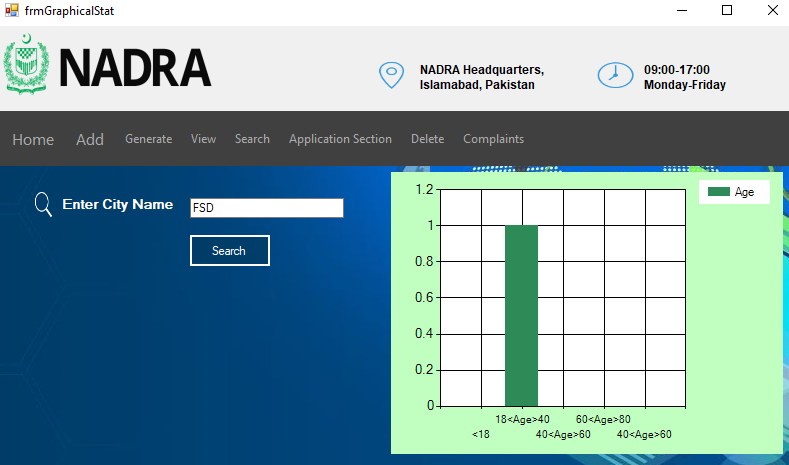


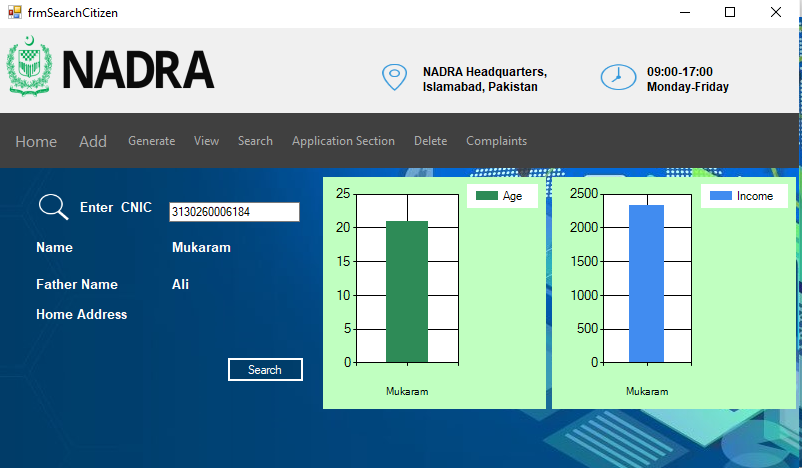


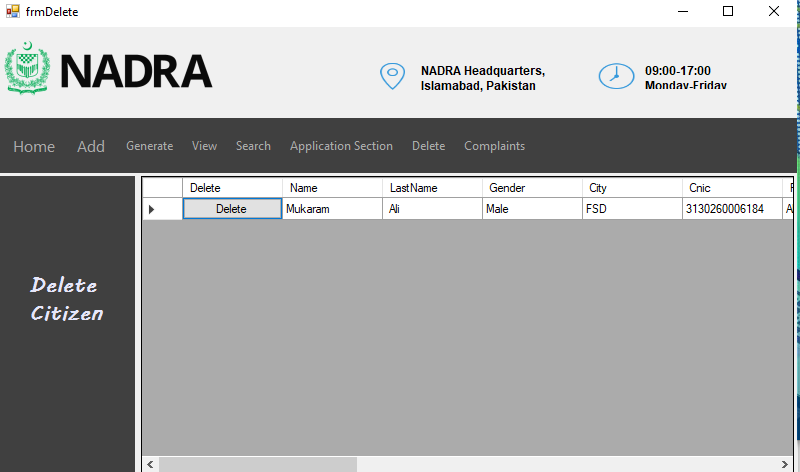


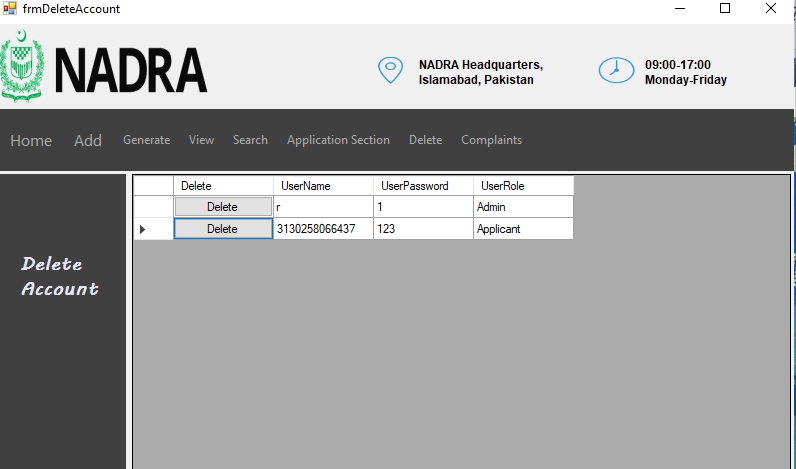


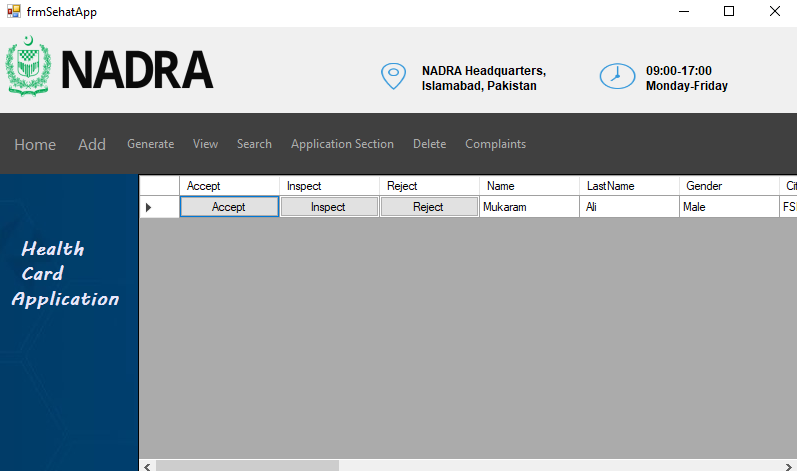


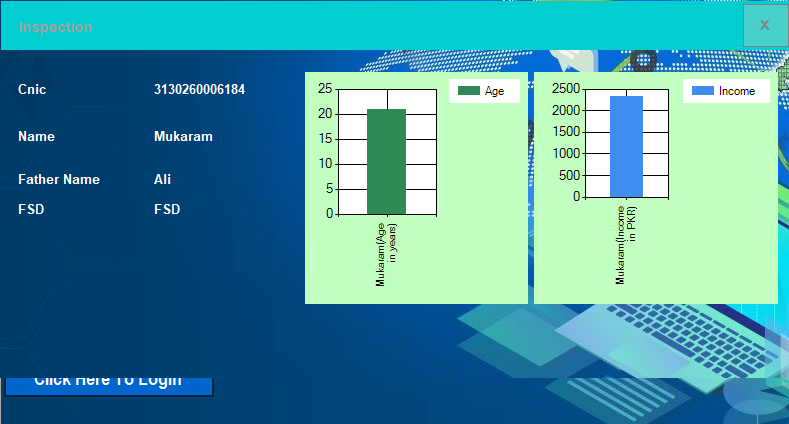




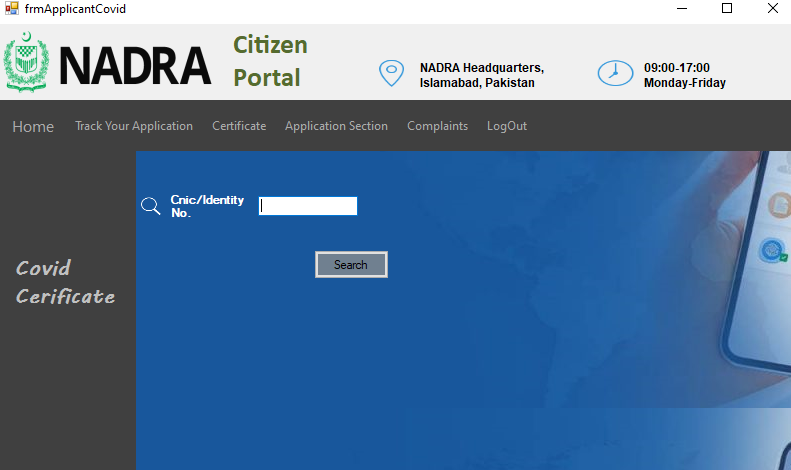




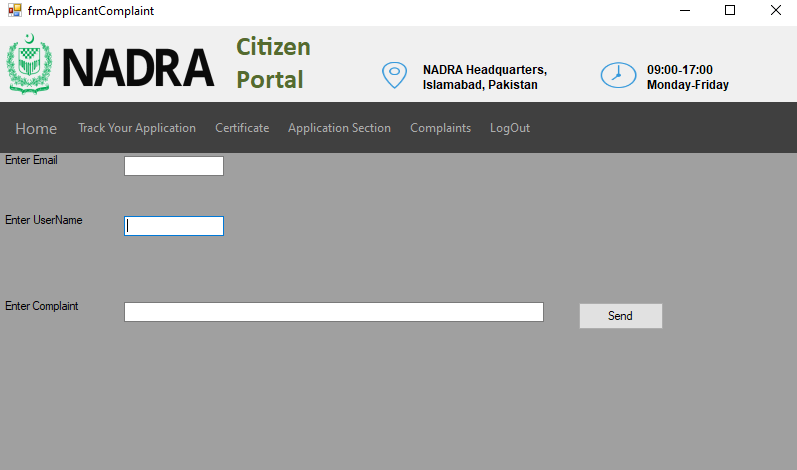


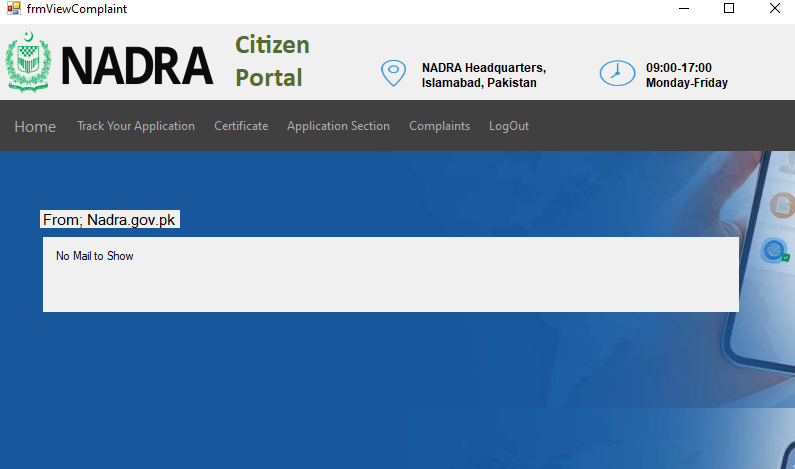












**CitizenBL**

public class citizen

{

private string name;

private string lastname;

private string gender;

private string city;

private string cnic;

private string fatherName;

private string province;

private string tempAdress;

private string permanentAdress;

private string vaccineName;

private int dose;

private int age;

private int year;

private int month;

private int date;

private int income;

private double worthTotal ;

private int tokenNumber;

public citizen(string name, string lastname, string gender, string city, string cnic, string fatherName, string province, string tempAdress, string permanentAdress, string vaccineName, int dose, int date, int month, int year, int income, double worthTotal)

{

this.name = name;

this.lastname = lastname;

this.gender = gender;

this.city = city;

this.cnic = cnic;

this.fatherName = fatherName;

this.province = province;

this.tempAdress = tempAdress;

this.permanentAdress = permanentAdress;

this.vaccineName = vaccineName;

this.dose = dose;

this.year = year;

this.month = month;

this.date = date;

this.income = income;

this.worthTotal = worthTotal;

}

public citizen(){

}

public string Name { get => name; set => name = value; }

public string LastName { get => lastname; set => lastname = value; }

public string Gender { get => gender; set => gender = value; }

public string City { get => city; set => city = value; }

public string Cnic { get => cnic; set => cnic = value; }

public string FatherName { get => fatherName; set => fatherName = value; }

public string Province { get => province; set => province = value; }

public string Temp\_adress { get => tempAdress; set => tempAdress = value; }

public string PermanentAdress { get => permanentAdress; set => permanentAdress = value; }

public string VaccineName { get => vaccineName; set => vaccineName = value; }

public int Dose { get => dose; set => dose = value; }

public int Age { get => age; set => age = value; }

public int TokenNumber { get => tokenNumber; set => tokenNumber = value; }

public int Year { get => year; set => year = value; }

public int Month { get => month; set => month = value; }

public int Date { get => date; set => date = value; }

public int Income { get => income; set => income = value; }

public double WorthTotal { get => worthTotal; set => worthTotal = value; }

}

**ComplaintBL**

public class Complaint

{

private string email;

private string username;

private string complaint;

private string reply;

public Complaint(string email,string username,string complaint)

{

this.email = email;

this.username = username;

this.complaint = complaint;

}

public string Name { get => username; set => username = value; }

public string Complaints { get => complaint; set => complaint = value; }

public string Email { get => email; set => email = value; }

public string Reply { get => reply; set => reply = value; }

}

**MUserBL**

public class MUser

{

private string userName;

private string userPassword;

private string userRole;

public string UserName { get => userName; set => userName = value; }

public string UserPassword { get => userPassword; set => userPassword = value; }

public string UserRole { get => userRole; set => userRole = value; }

public MUser(string userName, string userPassword, string userRole)

{

this.userName = userName;

this.userPassword = userPassword;

this.userRole = userRole;

}

public MUser()

{

}

public MUser(string userName, string userPassword)

{

this.userName = userName;

this.userPassword = userPassword;

this.userRole = "NA";

}

public bool isAdmin()

{

if (userRole == "Admin")

{

return true;

}

return false;

}

public static MUser SignIn(MUser user)

{

foreach (MUser storedUser in MUserCRUD.usersList)

{

if (storedUser.userName == user.userName && storedUser.userPassword == user.userPassword)

{

return storedUser;

}

}

return null;

}

}

class citizenCRUD

{

private static List<citizen> dataList = new List<citizen>();

private static List<citizen> sortedDataList = new List<citizen>();

private static List<citizen> sahatAppList = new List<citizen>();

private static List<citizen> acceptSahatAppList = new List<citizen>();

public static List<citizen> DataList { get => dataList; }

public static void addCitizenIntoList(citizen c)

{

dataList.Add(c);

}

public static void addSahatSelectedCitizen(citizen c)

{

acceptSahatAppList.Add(c);

}

public static void addsahatAppList(citizen c)

{

sahatAppList.Add(c);

}

static public void sortedDatalist()

{

sortedDataList = dataList.OrderByDescending(o => o.Age).ToList();

}

public static List<citizen> SortedDataList

{

get => sortedDataList;

}

public static List<citizen> AcceptSahatAppList

{

get => acceptSahatAppList;

}

public static List<citizen> SahatAppList

{

get => sahatAppList;set => sahatAppList = value;

}

public static void storeDataIntoFile(string path)

{

StreamWriter file = new StreamWriter(path, false);

for (int i = 0; i < dataList.Count; i++)

{

if (i == dataList.Count - 1)

{

file.Write(dataList[i].Name + ", " + dataList[i].LastName + "," + dataList[i].Gender + "," + dataList[i].City + "," + dataList[i].Cnic + "," + dataList[i].FatherName + "," + dataList[i].Province + "," + dataList[i].Temp\_adress + "," + dataList[i].PermanentAdress + "," + dataList[i].VaccineName + "," + dataList[i].Dose + ","+ dataList[i].Date + ","+ dataList[i].Month + ","+ dataList[i].Year + "," + dataList[i].Income + "," + dataList[i].WorthTotal+","+dataList[i].Age+","+ dataList[i].TokenNumber);

}

else

{

file.WriteLine(dataList[i].Name + ", " + dataList[i].LastName + "," + dataList[i].Gender + "," + dataList[i].City + "," + dataList[i].Cnic + "," + dataList[i].FatherName + "," + dataList[i].Province + "," + dataList[i].Temp\_adress + "," + dataList[i].PermanentAdress + "," + dataList[i].VaccineName + "," + dataList[i].Dose + "," + dataList[i].Date + "," + dataList[i].Month + "," + dataList[i].Year + "," + dataList[i].Income + "," + dataList[i].WorthTotal + "," + dataList[i].Age + "," + dataList[i].TokenNumber);

}

}

file.Flush();

file.Close();

}

public static void load( string path)

{

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*laoding recorda from file\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

string line;

if (File.Exists(path))

{

StreamReader file = new StreamReader(path);

while (((line = file.ReadLine())) != null)

{

string[] record = line.Split(',');

citizen Add = new citizen(record[0], record[1], record[2], record[3], record[4], record[5], record[6], record[7], record[8], record[9], int.Parse(record[10]), int.Parse(record[11]), int.Parse(record[12]), int.Parse(record[13]), int.Parse(record[14]), double.Parse(record[15]));

Add.Age = int.Parse(record[16]);

Add.TokenNumber = int.Parse(record[17]);

citizenCRUD.addCitizenIntoList(Add);

// uploaading temporary arr1ay data into orignal array

}

file.Close();

}

}

public static citizen isCitizenExist(string cnic)

{

foreach(var person in dataList)

{

if (person.Cnic == cnic)

{

return person;

}

}

return null;

}

static public void deleteUserFromList(citizen user,List<citizen >person)

{

for (int x = 0; x < person.Count; x++)

{

if (person[x].Cnic==user.Cnic)

{

person.RemoveAt(x);

}

}

}

public static void StoreIntoFileSahatApp(string path, string path1)

{

SahatAppStoreAppCitizen(path);

SahatAppStoreSelected(path1);

}

private static void SahatAppStoreAppCitizen(string path)

{

StreamWriter file = new StreamWriter(path, false);

for (int i = 0; i < sahatAppList.Count; i++)

{

if (i == sahatAppList.Count - 1)

{

file.Write(sahatAppList[i].Cnic);

}

else

{

file.WriteLine(sahatAppList[i].Cnic);

}

}

file.Flush();

file.Close();

}

private static void SahatAppStoreSelected(string path)

{

StreamWriter file = new StreamWriter(path, false);

for (int i = 0; i < acceptSahatAppList.Count; i++)

{

if (i == acceptSahatAppList.Count - 1)

{

file.Write(acceptSahatAppList[i].Cnic);

}

else

{

file.WriteLine(acceptSahatAppList[i].Cnic);

}

}

file.Flush();

file.Close();

}

public static void loadFromFileSahatApp(string path,string path1)

{

SahatAppLoadAppCitizen(path);

SahatAppLoadSelected(path1);

}

private static void SahatAppLoadAppCitizen(string path)

{

string line;

if (File.Exists(path))

{

StreamReader file = new StreamReader(path);

while (((line = file.ReadLine())) != null)

{

citizen person = isCitizenExist(line);

addsahatAppList(person);

}

file.Close();

}

}

private static void SahatAppLoadSelected(string path)

{

string line;

if (File.Exists(path))

{

StreamReader file = new StreamReader(path);

while (((line = file.ReadLine())) != null)

{

citizen person = isCitizenExist(line);

addSahatSelectedCitizen(person);

}

file.Close();

}

}

public static citizen citizenFromTokenNumber(int TokenNumber)

{

foreach (var person in dataList)

{

if (person.TokenNumber == TokenNumber)

{

return person;

}

}

return null;

}

}

class ComplaintCRUD

{

private static List<Complaint> complaintList = new List<Complaint>();

public static List<Complaint> ComplaintList

{

get => complaintList;

set => complaintList = value;

}

public static void addIntoList(Complaint c)

{

complaintList.Add(c);

}

public static void storeDataIntoFile(string path)

{

StreamWriter file = new StreamWriter(path, false);

for (int i = 0; i < ComplaintList.Count; i++)

{

if (i == ComplaintList.Count - 1)

{

file.Write(ComplaintList[i].Email + "," + ComplaintList[i].Name + ",", ComplaintList[i].Complaints);

}

else

{

file.WriteLine(ComplaintList[i].Email + "," + ComplaintList[i].Name + "," + ComplaintList[i].Complaints);

}

}

file.Flush();

file.Close();

}

public static void load(string path)

{

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*laoding recorda from file\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

string line;

if (File.Exists(path))

{

StreamReader file = new StreamReader(path);

while (((line = file.ReadLine())) != null)

{

string[] record = line.Split(',');

Complaint Add = new Complaint(record[0], record[1], record[2]);

addIntoList(Add);

// uploaading temporary arr1ay data into orignal array

}

file.Close();

}

}

public static Complaint getSpecified(MUser user)

{

foreach(var cmp in complaintList)

{

if (cmp.Name == user.UserName)

{

return cmp;

}

}

return null;

}

}

class MUserCRUD

{

public static List<MUser> usersList = new List<MUser>();

public static void addUserIntoList(MUser user)

{

usersList.Add(user);

}

public static List<MUser> UsersList { get => usersList; }

public static bool readDataFromFile(string path)

{

if (File.Exists(path))

{

StreamReader fileVariable = new StreamReader(path);

string record;

while ((record = fileVariable.ReadLine()) != null)

{

string userName = parseData(record, 1);

string userPassword = parseData(record, 2);

string userRole = parseData(record, 3);

MUser user = new MUser(userName, userPassword, userRole);

addUserIntoList(user);

}

fileVariable.Close();

return true;

}

return false;

}

static string parseData(string record, int field)

{

int comma = 1;

string item = "";

for (int x = 0; x < record.Length; x++)

{

if (record[x] == ',')

{

comma++;

}

else if (comma == field)

{

item = item + record[x];

}

}

return item;

}

static public void storeUserIntoFile( string path)

{

//StreamWriter file = new StreamWriter(path, true);

//file.WriteLine(user.UserName+ "," + user.UserPassword + "," + user.UserRole);

//file.Flush();

//file.Close();

StreamWriter file = new StreamWriter(path);

foreach (MUser storedUser in usersList)

{

file.WriteLine(storedUser.UserName + "," + storedUser.UserPassword + "," + storedUser.UserRole);

}

file.Flush();

file.Close();

}

static public MUser checkUserValidation(string userName,string password)

{

foreach(var user in usersList)

{

if (user.UserName == userName && user.UserPassword == password)

{

return user;

}

}

return null;

}

static public void deleteUserFromList(MUser user)

{

for (int x = 0; x < usersList.Count; x++)

{

if (usersList[x].UserName == user.UserName && usersList[x].UserPassword == user.UserPassword && usersList[x].UserRole == user.UserRole)

{

usersList.RemoveAt(x);

}

}

}

class MUserCRUD

{

public static List<MUser> usersList = new List<MUser>();

public static void addUserIntoList(MUser user)

{

usersList.Add(user);

}

public static List<MUser> UsersList { get => usersList; }

public static bool readDataFromFile(string path)

{

if (File.Exists(path))

{

StreamReader fileVariable = new StreamReader(path);

string record;

while ((record = fileVariable.ReadLine()) != null)

{

string userName = parseData(record, 1);

string userPassword = parseData(record, 2);

string userRole = parseData(record, 3);

MUser user = new MUser(userName, userPassword, userRole);

addUserIntoList(user);

}

fileVariable.Close();

return true;

}

return false;

}

static string parseData(string record, int field)

{

int comma = 1;

string item = "";

for (int x = 0; x < record.Length; x++)

{

if (record[x] == ',')

{

comma++;

}

else if (comma == field)

{

item = item + record[x];

}

}

return item;

}

static public void storeUserIntoFile( string path)

{

//StreamWriter file = new StreamWriter(path, true);

//file.WriteLine(user.UserName+ "," + user.UserPassword + "," + user.UserRole);

//file.Flush();

//file.Close();

StreamWriter file = new StreamWriter(path);

foreach (MUser storedUser in usersList)

{

file.WriteLine(storedUser.UserName + "," + storedUser.UserPassword + "," + storedUser.UserRole);

}

file.Flush();

file.Close();

}

static public MUser checkUserValidation(string userName,string password)

{

foreach(var user in usersList)

{

if (user.UserName == userName && user.UserPassword == password)

{

return user;

}

}

return null;

}

static public void deleteUserFromList(MUser user)

{

for (int x = 0; x < usersList.Count; x++)

{

if (usersList[x].UserName == user.UserName && usersList[x].UserPassword == user.UserPassword && usersList[x].UserRole == user.UserRole)

{

usersList.RemoveAt(x);

}

}

}

class MUserCRUD

{

public static List<MUser> usersList = new List<MUser>();

public static void addUserIntoList(MUser user)

{

usersList.Add(user);

}

public static List<MUser> UsersList { get => usersList; }

public static bool readDataFromFile(string path)

{

if (File.Exists(path))

{

StreamReader fileVariable = new StreamReader(path);

string record;

while ((record = fileVariable.ReadLine()) != null)

{

string userName = parseData(record, 1);

string userPassword = parseData(record, 2);

string userRole = parseData(record, 3);

MUser user = new MUser(userName, userPassword, userRole);

addUserIntoList(user);

}

fileVariable.Close();

return true;

}

return false;

}

static string parseData(string record, int field)

{

int comma = 1;

string item = "";

for (int x = 0; x < record.Length; x++)

{

if (record[x] == ',')

{

comma++;

}

else if (comma == field)

{

item = item + record[x];

}

}

return item;

}

static public void storeUserIntoFile( string path)

{

//StreamWriter file = new StreamWriter(path, true);

//file.WriteLine(user.UserName+ "," + user.UserPassword + "," + user.UserRole);

//file.Flush();

//file.Close();

StreamWriter file = new StreamWriter(path);

foreach (MUser storedUser in usersList)

{

file.WriteLine(storedUser.UserName + "," + storedUser.UserPassword + "," + storedUser.UserRole);

}

file.Flush();

file.Close();

}

static public MUser checkUserValidation(string userName,string password)

{

foreach(var user in usersList)

{

if (user.UserName == userName && user.UserPassword == password)

{

return user;

}

}

return null;

}

static public void deleteUserFromList(MUser user)

{

for (int x = 0; x < usersList.Count; x++)

{

if (usersList[x].UserName == user.UserName && usersList[x].UserPassword == user.UserPassword && usersList[x].UserRole == user.UserRole)

{

usersList.RemoveAt(x);

}

}

}

* **HelperUI.cs**

public partial class frmAddCitizen : Form

{

public frmAddCitizen()

{

InitializeComponent();

}

private void cmdAdd\_Click(object sender, EventArgs e)

{

if (txtFName.Text!=""&& txtLastName.Text != ""&& cboGender.Text != ""&& txtCity.Text != ""&& txtCnic.Text != ""&& txtFatherName.Text != ""&& cboProvince.Text != ""&& txtTempAdress.Text != ""&& txtPermAdress.Text != ""&& cboVaccine.Text != ""&& cboDose.Text != ""&&dateTimePicker1.Text != ""&& txtIncome.Text != ""&& txtTotalWorth.Text != "")

{

int year = dateTimePicker1.Value.Year;

int month = dateTimePicker1.Value.Month;

int day = dateTimePicker1.Value.Day;

int Presentyear = DateTime.Now.Year;

citizen Add = new citizen(txtFName.Text, txtLastName.Text, cboGender.Text, txtCity.Text, txtCnic.Text, txtFatherName.Text, cboProvince.Text, txtTempAdress.Text, txtPermAdress.Text, cboVaccine.Text, int.Parse(cboDose.Text), day, month, year, int.Parse(txtIncome.Text), int.Parse(txtTotalWorth.Text));

Add.Age = Presentyear - year;

citizenCRUD.addCitizenIntoList(Add);

citizenCRUD.storeDataIntoFile(FilePath.dataPath);

Admin a = new Admin();

a.Show();

this.Hide();

}

else

{

MessageBox.Show("Make sure all field should be filled!");

}

}

private void userLoginToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmAddUser form = new frmAddUser();

form.Show();

}

private void txtFName\_TextChanged(object sender, EventArgs e)

{

}

private void tableLayoutPanel1\_Paint(object sender, PaintEventArgs e)

{

}

private void searchToolStripMenuItem\_Click(object sender, EventArgs e)

{

public partial class frmAddUser : Form

{

public frmAddUser()

{

InitializeComponent();

}

private void cmdClose\_Click(object sender, EventArgs e)

{

this.Close();

}

private void cmdSignUp\_Click(object sender, EventArgs e)

{

MUser user = new MUser(txtUserName.Text, txtPassword.Text, cboUserRole.Text);

MUserCRUD.addUserIntoList(user);

MUserCRUD.storeUserIntoFile(FilePath.credentialPath);

MessageBox.Show("SuccessFully Done!");

this.Close();

}

}

public partial class frmAddUser : Form

{

public frmAddUser()

{

InitializeComponent();

}

private void cmdClose\_Click(object sender, EventArgs e)

{

this.Close();

}

private void cmdSignUp\_Click(object sender, EventArgs e)

{

MUser user = new MUser(txtUserName.Text, txtPassword.Text, cboUserRole.Text);

MUserCRUD.addUserIntoList(user);

MUserCRUD.storeUserIntoFile(FilePath.credentialPath);

MessageBox.Show("SuccessFully Done!");

this.Close();

}

}

public partial class frmApplicantComplaint : frmApplicant

{

public frmApplicantComplaint()

{

InitializeComponent();

}

private void tableLayoutPanel4\_Paint(object sender, PaintEventArgs e)

{

}

private void cmdSend\_Click(object sender, EventArgs e)

{

Complaint c = new Complaint(txtEmail.Text,txtUserName.Text, txtComplaint.Text);

ComplaintCRUD.addIntoList(c);

ComplaintCRUD.storeDataIntoFile(FilePath.complaintpath);

txtUserName.Text = "";

txtEmail.Text = "";

txtComplaint.Text="";

}

}

public partial class frmApplicantCovid : frmApplicant

{

public frmApplicantCovid()

{

InitializeComponent();

}

private void frmApplicantCovid\_Load(object sender, EventArgs e)

{

}

private void cmdSearch\_Click(object sender, EventArgs e)

{

citizen person = citizenCRUD.isCitizenExist(txtToken.Text);

if (person != null)

{

frmCovidCard form = new frmCovidCard(person);

form.Show();

}

else

{

MessageBox.Show("Not Registered!");

}

}

}

public partial class frmApplicantHealth : frmApplicant

{

public frmApplicantHealth()

{

InitializeComponent();

}

private void frmApplicantHealth\_Load(object sender, EventArgs e)

{

lblNameFront.Text = Communication.Person.Name;

}

private void button1\_Click(object sender, EventArgs e)

{

if (Communication.Person!=null){

citizenCRUD.SahatAppList.Add(Communication.Person);

}

}

}

public partial class frmCnic : Admin

{

public frmCnic()

{

InitializeComponent();

}

private void frmCnic\_Load(object sender, EventArgs e)

{

pnlCnic.Hide();

cmdPrint.Hide();

}

private void button1\_Click(object sender, EventArgs e)

{

pnlCnic.Show();

cmdPrint.Show();

if (int.Parse(txtToken.Text) > 1000 && int.Parse(txtToken.Text) < 2000) {

citizen person = citizenCRUD.citizenFromTokenNumber(int.Parse(txtToken.Text));

if (person != null)

{

lblName.Text = person.Name;

lblFatherName.Text = person.FatherName;

lblGender.Text = person.Gender;

lblCnicNo.Text = person.Cnic;

lblDOB.Text = $"{person.Date}-{person.Month}-{person.Year}";

}

else

{

MessageBox.Show("Invalid Token Number!");

}

}

}

private void cmdPrint\_Click(object sender, EventArgs e)

{

}

private void pnlCnic\_Paint(object sender, PaintEventArgs e)

{

}

private void cmdPrint\_Click\_1(object sender, EventArgs e)

{

PrintDialog d = new PrintDialog();

d.ShowDialog();

}

}

public partial class frmComplaint : Admin

{

public frmComplaint()

{

InitializeComponent();

}

public void dataBind()

{

gvComplaint.DataSource = null;

gvComplaint.DataSource = ComplaintCRUD.ComplaintList;

gvComplaint.Refresh();

}

private void frmComplaint\_Load(object sender, EventArgs e)

{

gvComplaint.DataSource = ComplaintCRUD.ComplaintList;

}

private void gvComplaint\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

Complaint cmp = (Complaint)gvComplaint.CurrentRow.DataBoundItem;

if (gvComplaint.Columns["Reply"].Index == e.ColumnIndex)

{

frmComplaintReply form = new frmComplaintReply(cmp);

form.Show();

}

}

}

* **MenuItemUI.cs**

public partial class frmComplaintReply : Form

{

Complaint comp;

public frmComplaintReply(Complaint comp)

{

this.comp = comp;

InitializeComponent();

}

private void frmComplaintReply\_Load(object sender, EventArgs e)

{

pnlReply.Hide();

lblEmail.Text = comp.Email;

lblComplaint.Text = comp.Complaints;

}

private void button1\_Click(object sender, EventArgs e)

{

pnlReply.Show();

}

private void cmdSend\_Click(object sender, EventArgs e)

{

comp.Reply = txtReply.Text;

pnlReply.Hide();

}

private void cmdClose\_Click(object sender, EventArgs e)

{

this.Close();

}

private void lblEmail\_Click(object sender, EventArgs e)

{

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

}

public partial class frmCovid : Admin

{

public frmCovid()

{

InitializeComponent();

}

private void cmdGenerate\_Click(object sender, EventArgs e)

{

citizen person = citizenCRUD.isCitizenExist(txtToken.Text);

if (person != null)

{

frmCovidCard form = new frmCovidCard(person);

form.Show();

}

else

{

MessageBox.Show("Invalid!");

}

}

public partial class frmCovidCard : Form

{

private citizen person;

public frmCovidCard(citizen person)

{

InitializeComponent();

this.person = person;

}

private void frmCovidCard\_Load(object sender, EventArgs e)

{

lblName.Text = person.Name;

lblCnic.Text = person.Cnic;

lblDOB.Text = $"{person.Date}-{person.Month}-{person.Year}";

}

private void cmdPrint\_Click(object sender, EventArgs e)

{

PrintDialog d = new PrintDialog();

d.ShowDialog();

}

private void pictureBox1\_MouseClick(object sender, MouseEventArgs e)

{

this.Close();

}

}

public partial class frmDelete : Form

{

public frmDelete()

{

InitializeComponent();

}

public void dataBind()

{

dataGridViewDeleteCitizen.DataSource = null;

dataGridViewDeleteCitizen.DataSource = citizenCRUD.SahatAppList;

dataGridViewDeleteCitizen.Refresh();

}

private void label12\_Click(object sender, EventArgs e)

{

}

private void tableLayoutPanel2\_Paint(object sender, PaintEventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

citizen person = (citizen)dataGridViewDeleteCitizen.CurrentRow.DataBoundItem;

if (dataGridViewDeleteCitizen.Columns["Delete"].Index == e.ColumnIndex)

{

citizenCRUD.deleteUserFromList(person, citizenCRUD.DataList);

citizenCRUD.storeDataIntoFile(FilePath.dataPath);

dataBind();

}

}

private void frmDelete\_Load(object sender, EventArgs e)

{

dataGridViewDeleteCitizen.DataSource = citizenCRUD.DataList;

}

private void userLoginToolStripMenuItem1\_Click(object sender, EventArgs e)

{

frmDeleteAccount form = new frmDeleteAccount();

form.Show();

this.Hide();

}

}

public partial class frmDeleteAccount : Admin

{

public frmDeleteAccount()

{

InitializeComponent();

}

public void dataBind()

{

dataGridViewDeleteAccount.DataSource = null;

dataGridViewDeleteAccount.DataSource = MUserCRUD.UsersList;

dataGridViewDeleteAccount.Refresh();

}

private void dataGridViewDeleteAccount\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

MUser person = (MUser)dataGridViewDeleteAccount.CurrentRow.DataBoundItem;

if (dataGridViewDeleteAccount.Columns["Delete"].Index == e.ColumnIndex)

{

MUserCRUD.deleteUserFromList(person);

MUserCRUD.storeUserIntoFile(FilePath.credentialPath);

dataBind();

}

}

private void userLoginToolStripMenuItem1\_Click(object sender, EventArgs e)

{

frmDeleteAccount form = new frmDeleteAccount();

form.Show();

this.Hide();

}

private void citizenToolStripMenuItem2\_Click(object sender, EventArgs e)

{

frmDelete form = new frmDelete();

form.Show();

this.Hide();

}

private void frmDeleteAccount\_Load(object sender, EventArgs e)

{

dataGridViewDeleteAccount.DataSource = MUserCRUD.UsersList;

}

}

public partial class frmGraphicalStat : Form

{

public frmGraphicalStat()

{

InitializeComponent();

}

private void applicantRecordToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmAddCitizen form = new frmAddCitizen();

form.Show();

this.Hide();

}

private void userLoginToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmAddUser form = new frmAddUser();

form.Show();

this.Hide();

}

private void sortedCitizenWithAgeToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmSortedCitizen form = new frmSortedCitizen();

form.Show();

this.Hide();

}

private void citizenToolStripMenuItem1\_Click(object sender, EventArgs e)

{

frmSearchCitizen form = new frmSearchCitizen();

form.Show();

this.Hide();

}

private void button1\_Click(object sender, EventArgs e)

{

int detect\_18 = 0;

int detect\_40 = 0;

int detect\_60 = 0;

int detect\_80 = 0;

int detect\_100 = 0;

// cout << ageA[0];

for (int i = 0; i < citizenCRUD.DataList.Count; i++)

{

if (citizenCRUD.DataList[i].City == txtCity.Text)

{

if ((citizenCRUD.DataList[i].Age <= 18))

{

detect\_18++;

}

if ((citizenCRUD.DataList[i].Age > 20 && citizenCRUD.DataList[i].Age <= 40))

{

detect\_40++;

}

if ((citizenCRUD.DataList[i].Age > 40 && citizenCRUD.DataList[i].Age <= 60))

{

detect\_60++;

}

if ((citizenCRUD.DataList[i].Age > 60 && citizenCRUD.DataList[i].Age <= 80))

{

detect\_80++;

}

if ((citizenCRUD.DataList[i].Age > 80))

{

detect\_100++;

}

}

}

chartAge.Series["Age"].Points.AddXY("<18", detect\_18);

chartAge.Series["Age"].Points.AddXY("18<Age>40", detect\_40);

chartAge.Series["Age"].Points.AddXY("40<Age>60", detect\_60);

chartAge.Series["Age"].Points.AddXY("60<Age>80", detect\_80);

chartAge.Series["Age"].Points.AddXY("40<Age>60", detect\_100);

}

private void exitToolStripMenuItem\_Click(object sender, EventArgs e)

{

}

private void recordToolStripMenuItem\_Click(object sender, EventArgs e)

{

Admin form = new Admin();

form.Show();

public partial class frmHealthInspect : Form

{

public frmHealthInspect()

{

InitializeComponent();

}

private void cmdClose\_Click(object sender, EventArgs e)

{

this.Close();

}

private void frmHealthInspect\_Load(object sender, EventArgs e)

{

citizen person=Communication.Person;

lblCnic.Text = "Cnic";

lblCnicFront.Text = person.Cnic;

lblName.Text = "Name";

lblNameFront.Text = person.Name;

lblFather.Text = "Father Name";

lblFatherFront.Text = person.FatherName;

lblAdress.Text = person.PermanentAdress;

lblAdressFront.Text = person.PermanentAdress;

chartSearchAge.Series["Age"].Points.AddXY($"{person.Name}(Age in years)",person.Age);

chartSearchIncome.Series["Income"].Points.AddXY($"{person.Name}(Income in PKR)",person.Income);

}

}

}

public partial class frmInvoice : Admin

{

public frmInvoice()

{

InitializeComponent();

}

private void cmdGenerate\_Click(object sender, EventArgs e)

{

int bill = 0;

citizen person=null;

if (int.Parse(txtToken.Text)>1000 && (int.Parse(txtToken.Text) < 2000))

{

person = citizenCRUD.citizenFromTokenNumber(int.Parse(txtToken.Text));

if (chkCnic.Checked)

{

if (rdUrgent.Checked)

{

bill = 2500;

lblAppFront.Text = "Urgent";

}

else if (rdRegular.Checked)

{

bill = 1500;

lblAppFront.Text = "Regular";

}

else

{

MessageBox.Show("Select Urgent or Regular");

}

}

}

else

{

MessageBox.Show("Invalid");

}

if (bill != 0)

{

lblName.Text = "Name:";

lblNameFront.Text = person.Name;

lblApp.Text = "Application Status:";

lblID.Text = "ID No.";

lblIDNo.Text = person.Cnic;

lblInvoice.Text = "Total Invoice";

lblNameFront.Text = $"{ bill}";

}

}

}

public partial class frmSearchCitizen : Form

{

public frmSearchCitizen()

{

InitializeComponent();

}

private void cmdSearch\_Click(object sender, EventArgs e)

{

citizen person=citizenCRUD.isCitizenExist(txtCity.Text);

if (person != null)

{

lblName.Text = "Name";

lblNameFront.Text = person.Name;

lblFather.Text = "Father Name";

//lblAdressFront.Text = person.Temp\_adress;

lblAdress.Text = "Home Address";

lblFatherFront.Text = person.FatherName;

chartSearchAge.Series["Age"].Points.AddXY(person.Name, person.Age);

chartSearchIncome.Series["Income"].Points.AddXY(person.Name, person.Income);

}

else

{

MessageBox.Show("No Citizen exist against this Cnic!");

}

}

this.Hide();

}

public partial class frmSehatApp : Form

{

public frmSehatApp()

{

InitializeComponent();

}

public void dataBind()

{

dataGridViewSahat.DataSource = null;

dataGridViewSahat.DataSource = citizenCRUD.SahatAppList;

dataGridViewSahat.Refresh();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

citizen person = (citizen)dataGridViewSahat.CurrentRow.DataBoundItem;

if (dataGridViewSahat.Columns["Accept"].Index == e.ColumnIndex)

{

//accpting and than delte from sahatapp list and add into selectedSahatList

citizenCRUD.addSahatSelectedCitizen(person);

citizenCRUD.deleteUserFromList(person,citizenCRUD.SahatAppList);

citizenCRUD.StoreIntoFileSahatApp(FilePath.sahatAppPath, FilePath.sahatSelectPath);

dataBind();

}

else if (dataGridViewSahat.Columns["Reject"].Index == e.ColumnIndex)

{

//delete application from sahatAppList

citizenCRUD.deleteUserFromList(person,citizenCRUD.SahatAppList);

citizenCRUD.StoreIntoFileSahatApp(FilePath.sahatAppPath, FilePath.sahatSelectPath);

dataBind();

}

else if (dataGridViewSahat.Columns["Inspect"].Index == e.ColumnIndex)

{

frmHealthInspect form = new frmHealthInspect();

Communication.Person = person;

form.Show();

}

}

private void frmSehatApp\_Load(object sender, EventArgs e)

{

citizenCRUD.sortedDatalist();

dataGridViewSahat.DataSource = citizenCRUD.SahatAppList;

}

private void citizenToolStripMenuItem1\_Click(object sender, EventArgs e)

{

frmSearchCitizen form = new frmSearchCitizen();

form.Show();

this.Hide();

}

private void sortedCitizenWithAgeToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmSortedCitizen form = new frmSortedCitizen();

form.Show();

this.Hide();

}

private void graphicalStatisticsOfCityToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmGraphicalStat form = new frmGraphicalStat();

form.Show();

this.Hide();

}

private void applicantRecordToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmAddCitizen form = new frmAddCitizen();

form.Show();

this.Hide();

}

public partial class frmSehatApp : Form

{

public frmSehatApp()

{

InitializeComponent();

}

public void dataBind()

{

dataGridViewSahat.DataSource = null;

dataGridViewSahat.DataSource = citizenCRUD.SahatAppList;

dataGridViewSahat.Refresh();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

citizen person = (citizen)dataGridViewSahat.CurrentRow.DataBoundItem;

if (dataGridViewSahat.Columns["Accept"].Index == e.ColumnIndex)

{

//accpting and than delte from sahatapp list and add into selectedSahatList

citizenCRUD.addSahatSelectedCitizen(person);

citizenCRUD.deleteUserFromList(person,citizenCRUD.SahatAppList);

citizenCRUD.StoreIntoFileSahatApp(FilePath.sahatAppPath, FilePath.sahatSelectPath);

dataBind();

}

else if (dataGridViewSahat.Columns["Reject"].Index == e.ColumnIndex)

{

//delete application from sahatAppList

citizenCRUD.deleteUserFromList(person,citizenCRUD.SahatAppList);

citizenCRUD.StoreIntoFileSahatApp(FilePath.sahatAppPath, FilePath.sahatSelectPath);

dataBind();

}

else if (dataGridViewSahat.Columns["Inspect"].Index == e.ColumnIndex)

{

frmHealthInspect form = new frmHealthInspect();

Communication.Person = person;

form.Show();

}

}

private void frmSehatApp\_Load(object sender, EventArgs e)

{

citizenCRUD.sortedDatalist();

dataGridViewSahat.DataSource = citizenCRUD.SahatAppList;

}

private void citizenToolStripMenuItem1\_Click(object sender, EventArgs e)

{

frmSearchCitizen form = new frmSearchCitizen();

form.Show();

this.Hide();

}

private void sortedCitizenWithAgeToolStripMenuItem\_Click(object sender, EventArgs e)

{

frmSortedCitizen form = new frmSortedCitizen();

form.Show();

this.Hide();

}

public partial class frmSortedCitizen : Form

{

public frmSortedCitizen()

{

InitializeComponent();

}

public void dataBind()

{

dataGridViewSorted.DataSource = null;

dataGridViewSorted.DataSource = citizenCRUD.SortedDataList;

dataGridViewSorted.Refresh();

}

private void tableLayoutPanel2\_Paint(object sender, PaintEventArgs e)

{

}

private void menuStrip\_ItemClicked(object sender, ToolStripItemClickedEventArgs e)

{

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void frmSortedCitizen\_Load(object sender, EventArgs e)

{

citizenCRUD.sortedDatalist();

dataGridViewSorted.DataSource = citizenCRUD.SortedDataList;

}

private void dataGridViewSorted\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void userLoginToolStripMenuItem1\_Click(object sender, EventArgs e)

{

frmDeleteAccount form = new frmDeleteAccount();

form.Show();

this.Hide();

}

public partial class frmTokenNo : Admin

{

public frmTokenNo()

{

InitializeComponent();

}

private void frmTokenNo\_Load(object sender, EventArgs e)

{

gvToken.DataSource = citizenCRUD.DataList;

}

public void dataBind()

{

gvToken.DataSource = null;

gvToken.DataSource =citizenCRUD.DataList;

gvToken.Refresh();

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

citizen person = (citizen)gvToken.CurrentRow.DataBoundItem;

if (gvToken.Columns["Generate"].Index == e.ColumnIndex)

{

Random r = new Random();

person.TokenNumber = r.Next(1000,2000);

citizenCRUD.storeDataIntoFile(FilePath.dataPath);

dataBind();

}

}

public partial class frmTrackApplication : frmApplicant

{

public frmTrackApplication()

{

InitializeComponent();

}

private void frmTrackApplication\_Load(object sender, EventArgs e)

{

pnlFind1.Hide();

pnlFind2.Hide();

}

private void cmdFind\_Click(object sender, EventArgs e)

{

pnlFind1.Show();

pnlFind2.Show();

}

private void cmdFinded\_Click(object sender, EventArgs e)

MessageBox.Show("Your tacking id is "+person.TokenNumber);

else

MessageBox.Show("No tracing ID found..Your application is in process");

}

else

{

MessageBox.Show("No registeres application found against this Cnic");

}

}

private void cmdtrack\_Click(object sender, EventArgs e)

{

if (int.Parse(txtTrack.Text) < 2000 && int.Parse(txtTrack.Text) > 1000)

{

citizen person = citizenCRUD.citizenFromTokenNumber(int.Parse(txtTrack.Text));

if (person != null)

{

MessageBox.Show(person.Name +"your application against tacking id " + person.TokenNumber+"is in completion mode.");

}

else

{

MessageBox.Show("Invalid!");

}

}

else

{

MessageBox.Show("Invalid!");

}

}

private void cmdFind\_Click\_1(object sender, EventArgs e)

{

pnlFind1.Show();

pnlFind2.Show();

}

}

}

public partial class frmViewComplaint : frmApplicant

{

public frmViewComplaint()

{

InitializeComponent();

}

private void frmViewComplaint\_Load(object sender, EventArgs e)

{

Complaint c = ComplaintCRUD.getSpecified(Communication.User);

if (c != null)

{

lblMail.Text = c.Reply;

}

}

private void lblMail\_Click(object sender, EventArgs e)

{

}

}

public partial class LoginScreen : Form

{

public LoginScreen()

{

InitializeComponent();

}

private void Form2\_Load(object sender, EventArgs e)

{

ComplaintCRUD.load(FilePath.complaintpath);

citizenCRUD.loadFromFileSahatApp(FilePath.sahatAppPath, FilePath.sahatSelectPath);

}

private void cmdSignIn\_Click(object sender, EventArgs e)

{

MUser user = MUserCRUD.checkUserValidation(txtUserName.Text, txtPassword.Text);

if (user != null)

{

if (user.UserRole == "Admin")

{

citizenCRUD.load(FilePath.dataPath);

Admin adminForm = new Admin();

adminForm.Show();

this.Hide();

}

else if (user.UserRole == "Applicant")

{

citizen c = citizenCRUD.isCitizenExist(txtUserName.Text);

Communication.Person = c;

Communication.User = user;

frmApplicant form = new frmApplicant();

form.Show();

this.Hide();

}

}

else

{

MessageBox.Show("Invalid Credentials");

}

}

private void cmdClose\_Click(object sender, EventArgs e)

{

this.Close();

}

private void cmdReset\_Click(object sender, EventArgs e)

{

txtPassword.Text = "";

txtUserName.Text = "";

}

private void lnkSignUp\_LinkClicked(object sender, LinkLabelLinkClickedEventArgs e)

{

SignUp form = new SignUp();

form.Show();

this.Hide();

}

private void txtUserName\_MouseClick(object sender, MouseEventArgs e)

{

if(txtUserName.Text== "Username")

txtUserName.Clear();

}

private void txtPassword\_MouseClick(object sender, MouseEventArgs e)

{

if (txtPassword.Text == "Password")

txtPassword.Clear();

}

}

}

public partial class SignUp : Form

{

private string path="credentials";

public SignUp()

{

InitializeComponent();

}

private void SignUp\_Load(object sender, EventArgs e)

{

}

private void cmdSignUp\_Click(object sender, EventArgs e)

{

MUser user = new MUser(txtUserName.Text, txtPassword.Text, cboUserRole.Text);

MUserCRUD.addUserIntoList(user);

MUserCRUD.storeUserIntoFile(FilePath.credentialPath);

this.Close();

}

private void cmdClose\_Click(object sender, EventArgs e)

{

this.Close();

}

private void txtUserName\_MouseClick(object sender, MouseEventArgs e)

{

if (txtUserName.Text == "Username")

txtUserName.Clear();

}

private void txtPassword\_MouseClick(object sender, MouseEventArgs e)

{

if (txtUserName.Text == "Password")

txtUserName.Clear();

}

}