



**Assuit university**  
**Faculty Of Computer And Information**  
**Science**

**Graduation Project Book**  
**HX Wallet**

## Contents



Assuit university ..... 0

1.	Project initiation:	2
1.1.1.	Team members and supervised	2
1.1.2.	Introduction	3
1.1.3.	Problem definition	4
1.1.4.	Litrature overview	5
1.1.5.	Objectives	6
1.1.6.	Stakeholders	7
1.1.7.	Scope	7
1.1.8.	Constraints	8
2.	Planning & Requirements:	8
2.1.1.	Cost and budget	8
2.1.2.	Risk list	9
2.1.3.	Requirements	9
3.	Analysis and Design:	12
3.1.1.	Use case	12
3.1.2.	Data Flow Diagram	13
3.1.3.	Flow Chart	16
3.1.4.	Activity	18
3.1.5.	Context	21
4.	Implementation:	24
4.1.1.	Development requirements	24
4.1.2.	Manual User	25
4.1.3.	Screens of code	34
4.1.3.	References	56
5.	Conclusion and Future Improvement:	56
5.1.1.	Conclusion	56
5.1.2.	Future Improvement	57

## **1. Project initiation:**

### **1.1.1. Team members and supervised**

**Project name:**

HX Wallet

**Team members:**

Neama Mamdoh Ramadan Abdelkader

Aya Mostafa Kamel Hashem

Entesar Atef Abd elrhman omar

Doaa Ahmed Salman Ahmed

Doaa Saad-Eldein Yassin

Yousef Elsaied Yousef Abdallah

Mohamed Mahmoud Abdeltawab

Yousef

**Supervised By:**

DR/Ebrahim El Semman

## 1.1.2. Introduction

The medical prescription is the means in which the doctor presents his prescription to the patient, in which he presents all the details of the treatment, including the name of the treatment, the dose to be taken, and the dates for taking the treatment

The presence of the prescription with the patient makes it easier for him to recover from the disease because he does not follow the instructions inside.

And because the medical prescription is very important in the patient's recovery journey, our project will focus on the prescription, but it is an electronic prescription.

### 1.1.3. Problem definition

- Imagine going to the doctor and taking his prescription, and after disbursing the treatment, you needed to repeat the medicines again, but unfortunately you lost the prescription or going to doctor and he asking you if you take a specific medicine to take it into account when writing a new prescription to you.
- So we trying to solve the lose of the prescription from patients.
- We are trying to solve this problem in our project through making a website that contain all patient prescriptions and tests

#### **1.1.4. Literature overview**

- This is not the first idea of creating an electronic prescription, there have been many attempts to create websites with the same idea
- for example NHS Digital website has an electronic prescription service in its services.
- But unfortunately this service does not exist in Egypt .

### 1.1.5. Objectives

- We aim to facilitate the patient's access to his old prescriptions, tests, x-rays and to cancel the idea of the paper prescription because of its many defects such as loss or damage.
- On the other hand, we aim to help the doctor know the medications that the patient is currently taking, and this will facilitate what he can put in his new prescription so as not to conflict with the current medications.
- in addition to cancel the paper tests results that printed on a papers and always patients lose it .
- for this reasons we made our website to help all of doctor ,patient,pharmacy and test and x-ray labs

## 1.1.6. Stakeholders

- We know previously that the stakeholders are those who are directly or indirectly affected by the project, so here our stakeholders are:
- The patient
- The doctor
- The pharmacy
- The test lab
- The x-ray lab

## 1.1.7. Scope

2. Our project is a website that helps you to keep your medical history in your profile, all your medicine , all your tests, all your X-rays in electronic page and get rid of all papers you may hold or lose them.

## **2.1.1. Constraints**

To use our website you need the following:

- A mobile phone, laptop, or computer device to access the website through it
- An internet connection
- A browser that reaches the website through

## **3. Planning & Requirements:**

### **3.1.1. Cost and budget**

- It is an educational project that we have done because it is one of the requirements for graduation and obtaining a graduation certificate and a bachelor's degree. It is also a very suitable start to prepare us for the work environment.
- So there is no development cost or cost for equipment

### 3.1.2. Risk list

- Server failure
- Internet failure
- The absence of mobile or laptop to browse through it

### 3.1.3. Requirements

#### functional requirements

- For all users:
  - Sign up:when the user enter the site for the first time he enter all his information and submit the information so he has a profile now and he can enter to it .
  - Login to the site:the user enter the website and enter his user name and password and reach to his page.
- For the patient:
  - **show his profile:**when the patient login to the site he show his profile

that contain all his prescriptions and tests and x-rays.

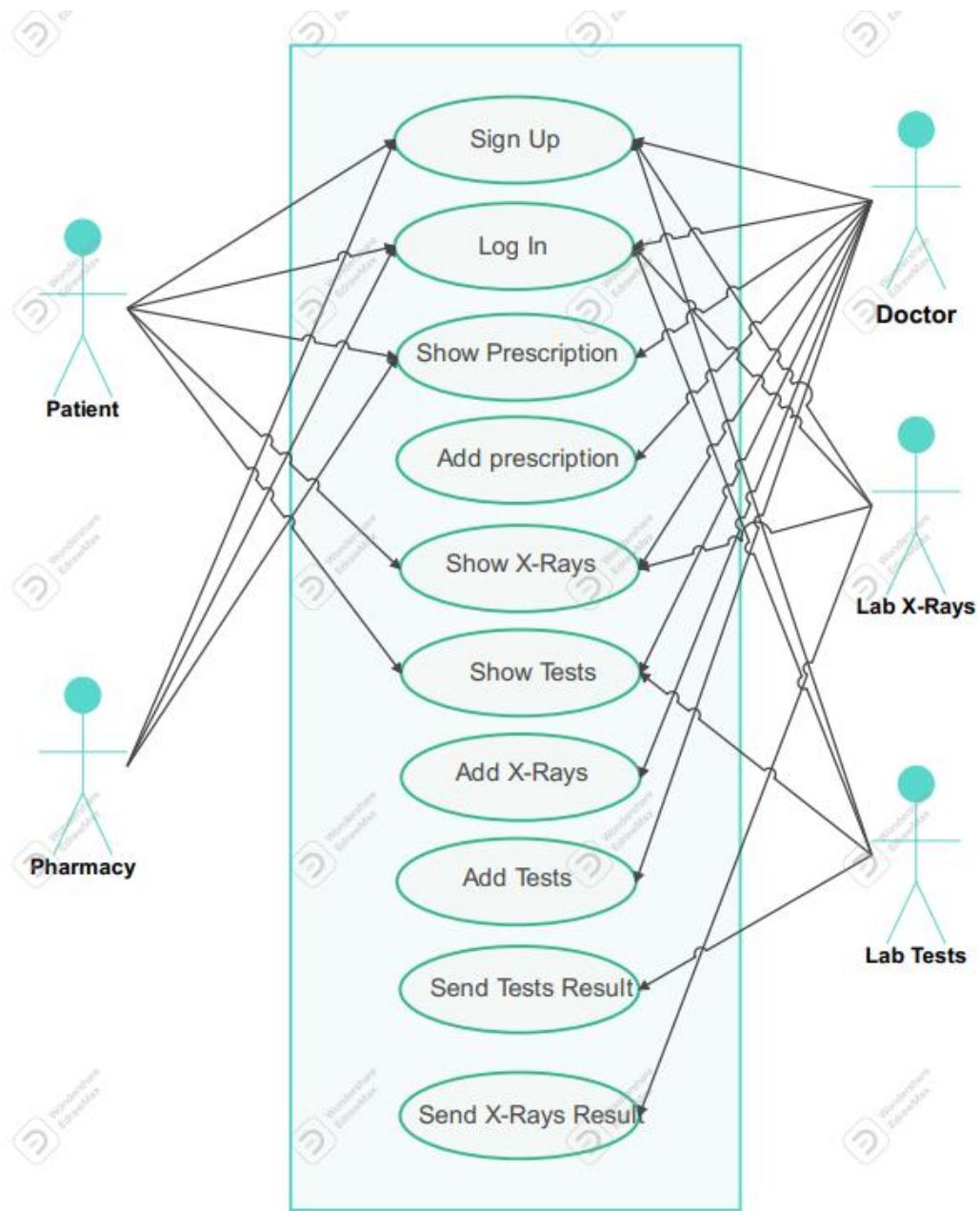
- For the doctor:
  - Show patient profile
  - Add new prescription
- For the pharmacy:
  - Show prescriptions of the patient
  - Mark the medication the patient has taken from them
- For test and x-ray lab:
  - Add tests to patients
  - Show all patient tests required from him

## **nonfunctional requirements:**

- For the patient:
  - Search on patient profile for prescription or tests
- For the doctor:
  - Search on patient profile for prescription or tests
- For the pharmacy:
  - Search for a specific medicine in a prescription
- For test and x-ray lab:
  - Search for specific test

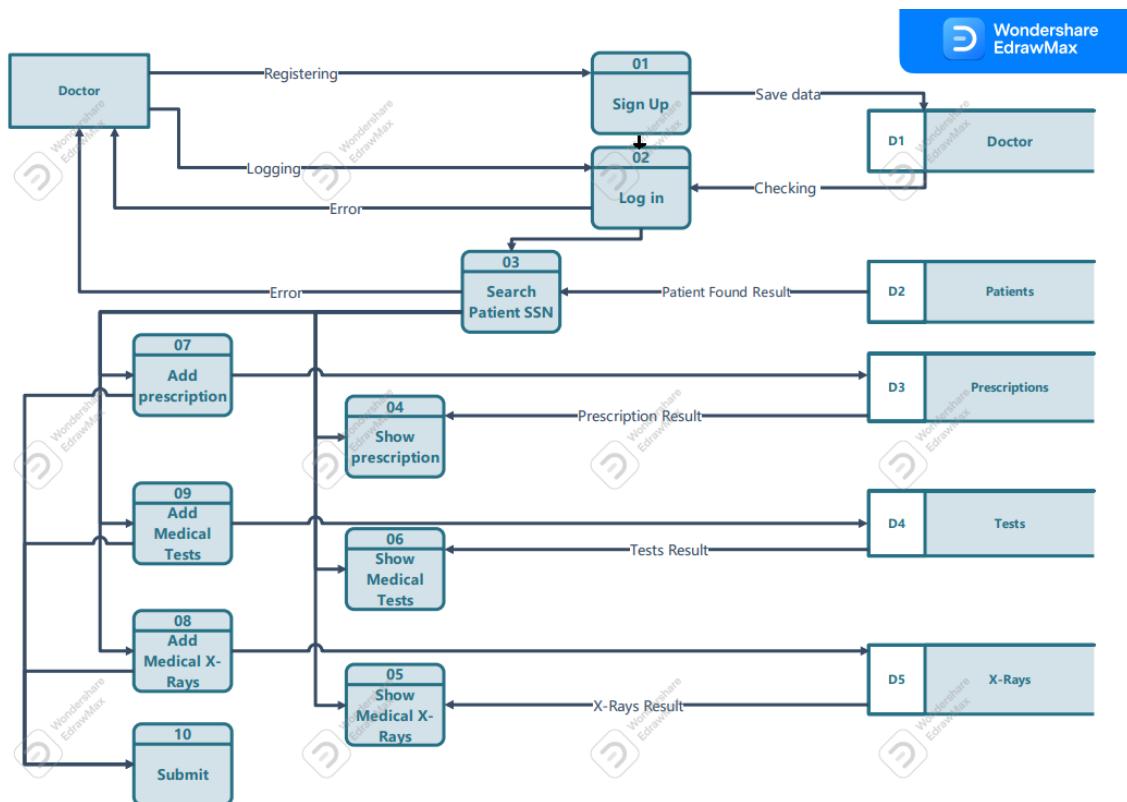
## 4. Analysis and Design:

### 4.1.1. Use case

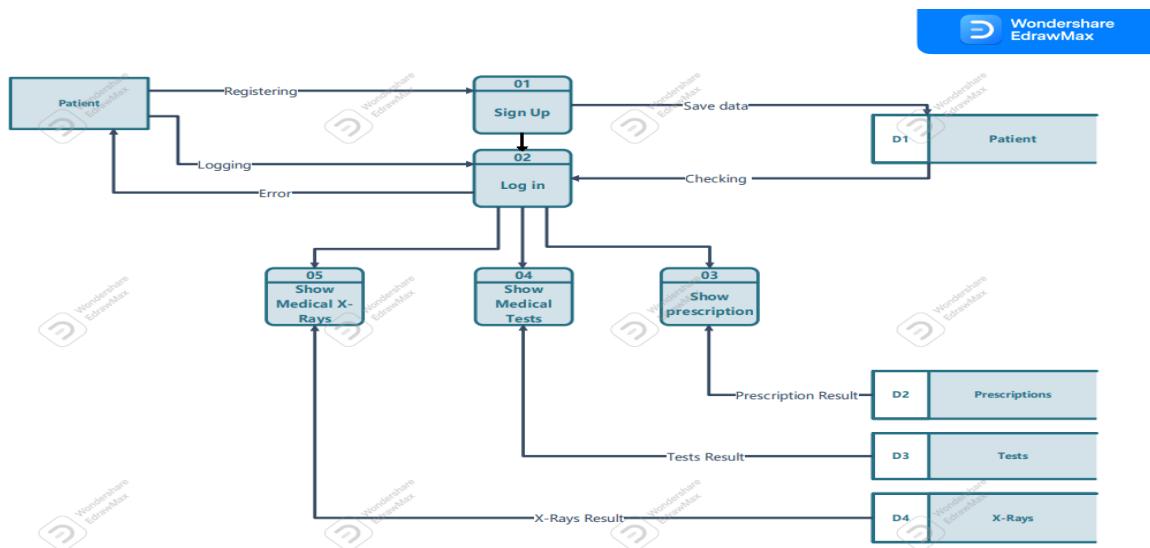


## 4.1.2. Data Flow Diagram

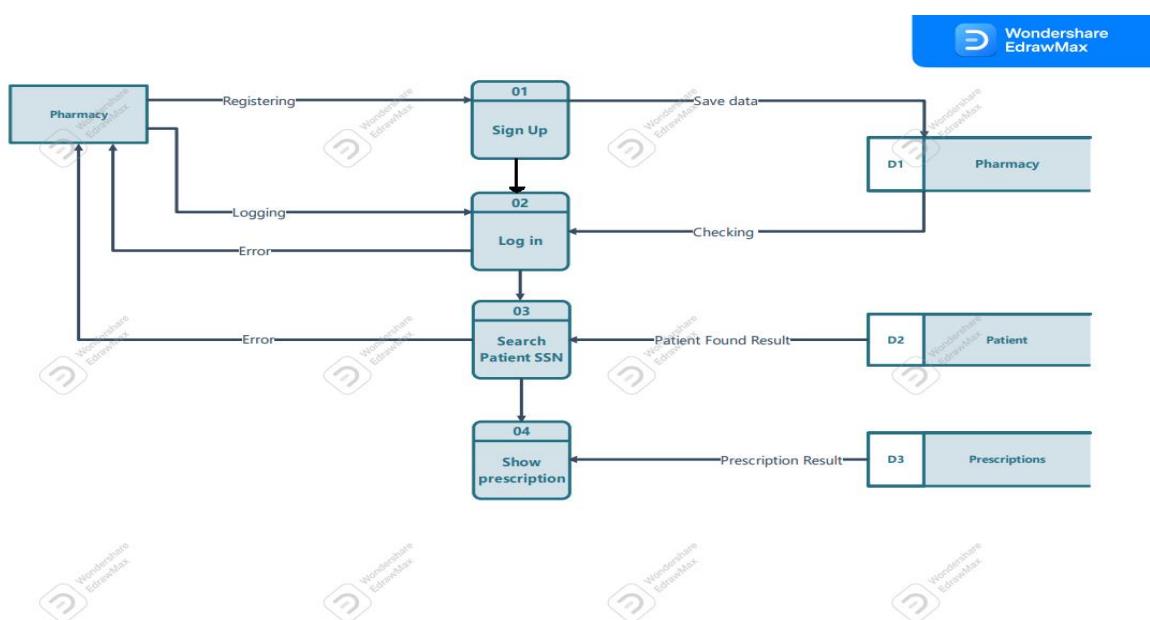
For doctor



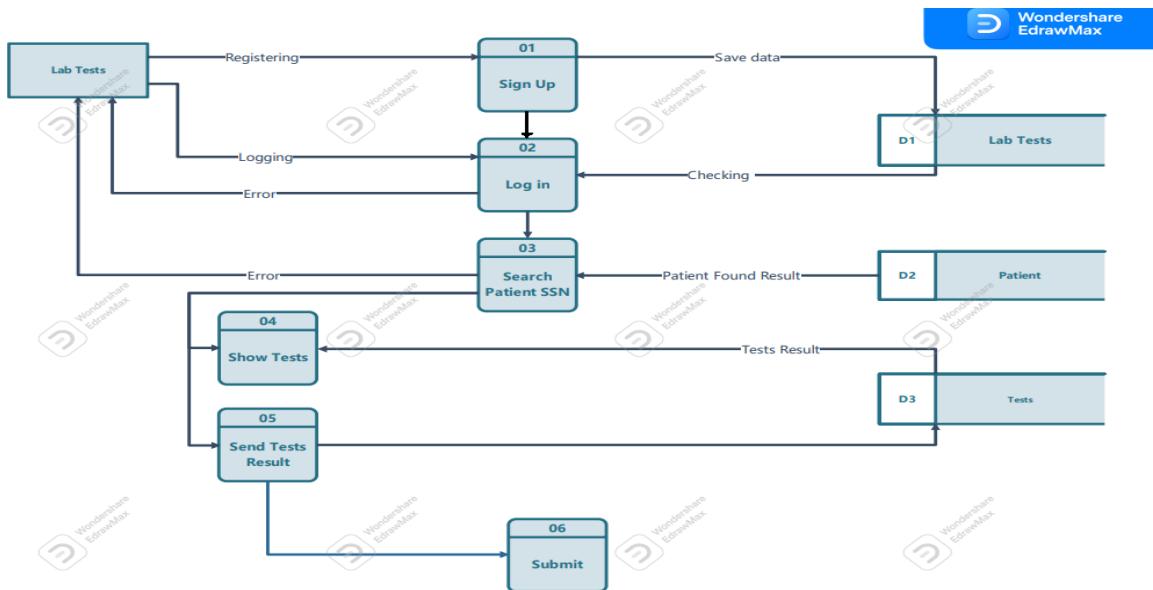
# For patient



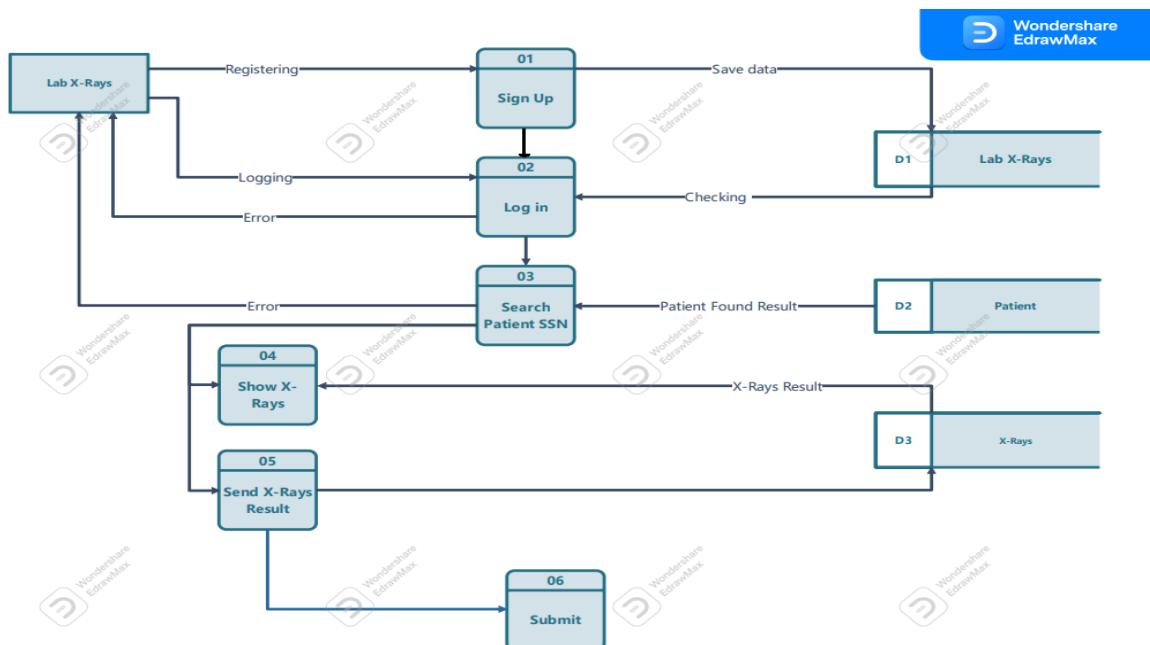
# For pharmacy



# For test lab

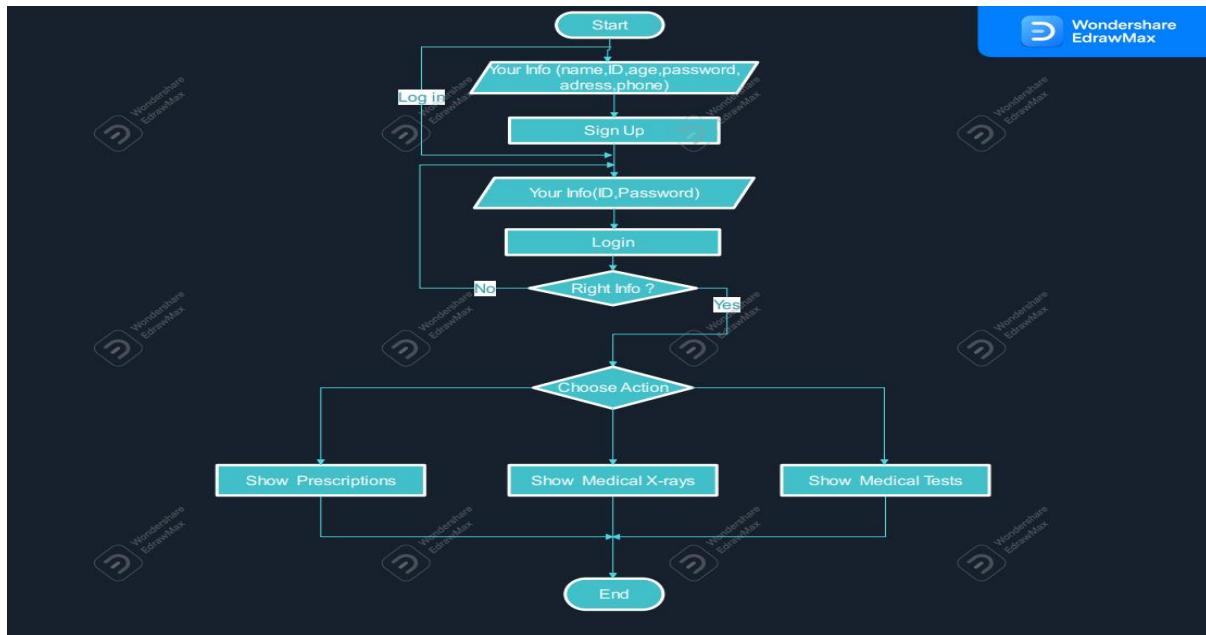


# For x-ray lab

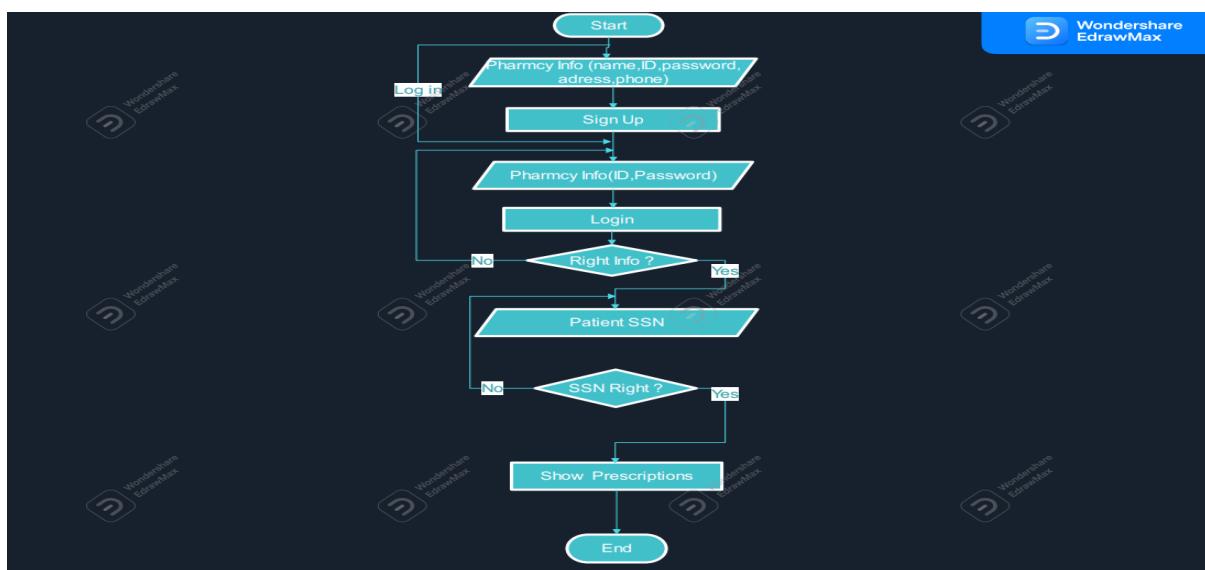


### 4.1.3. Flow Chart

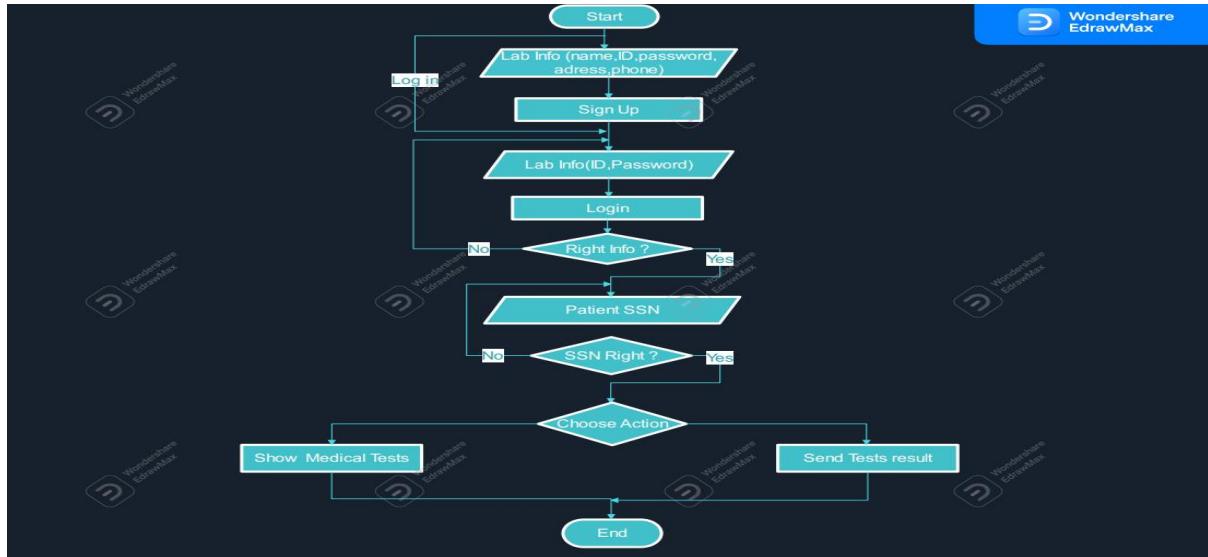
For doctor



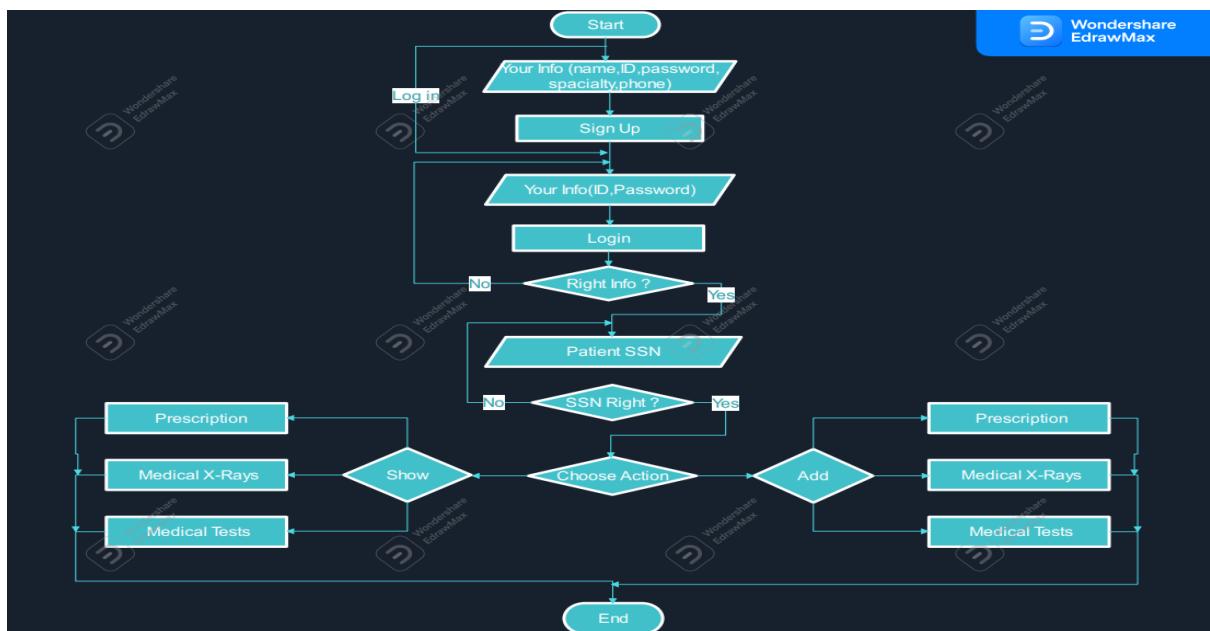
For patient



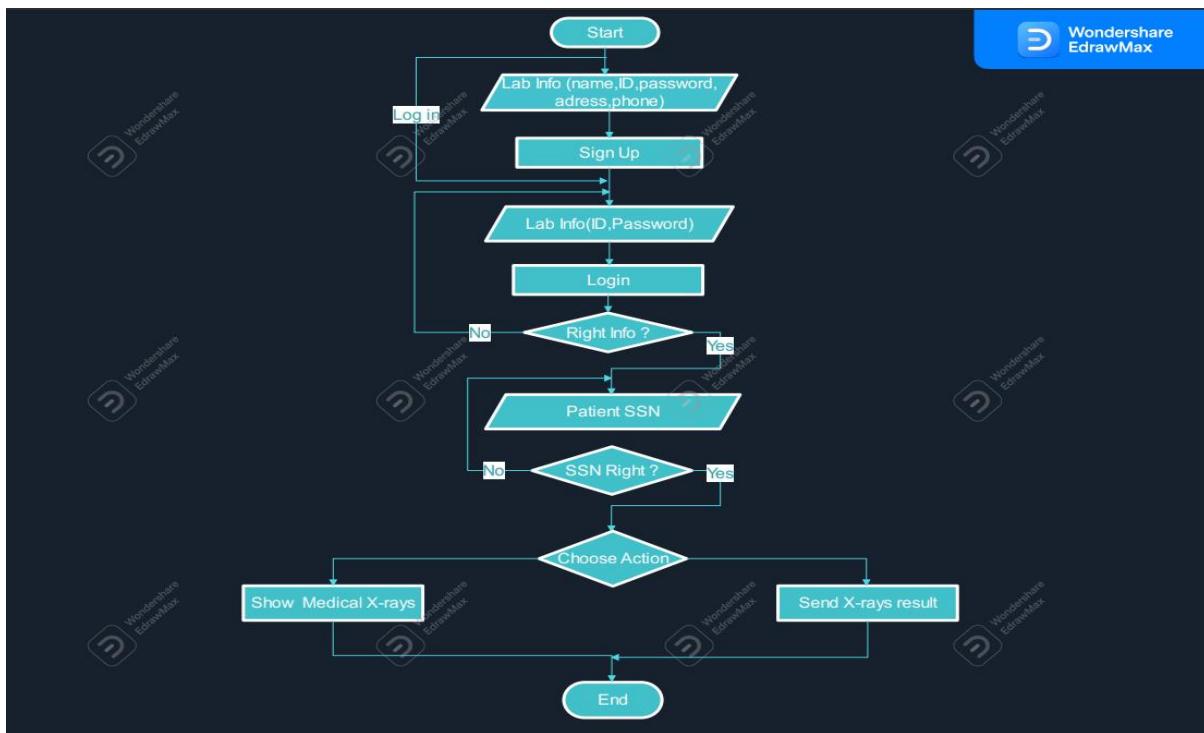
# For pharmacy



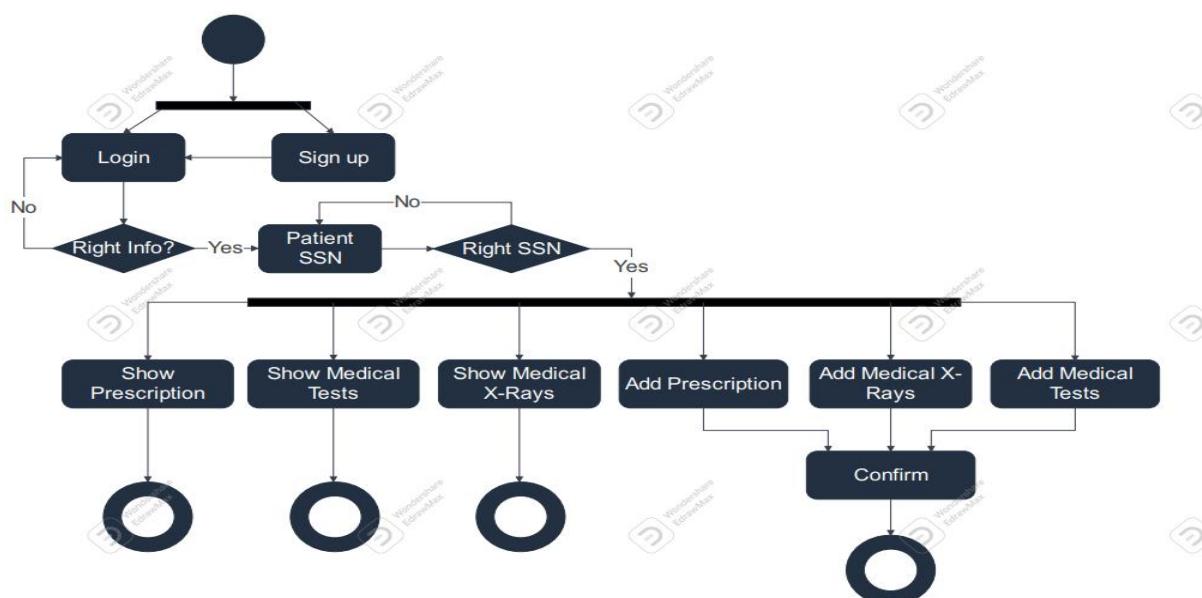
# For test lab



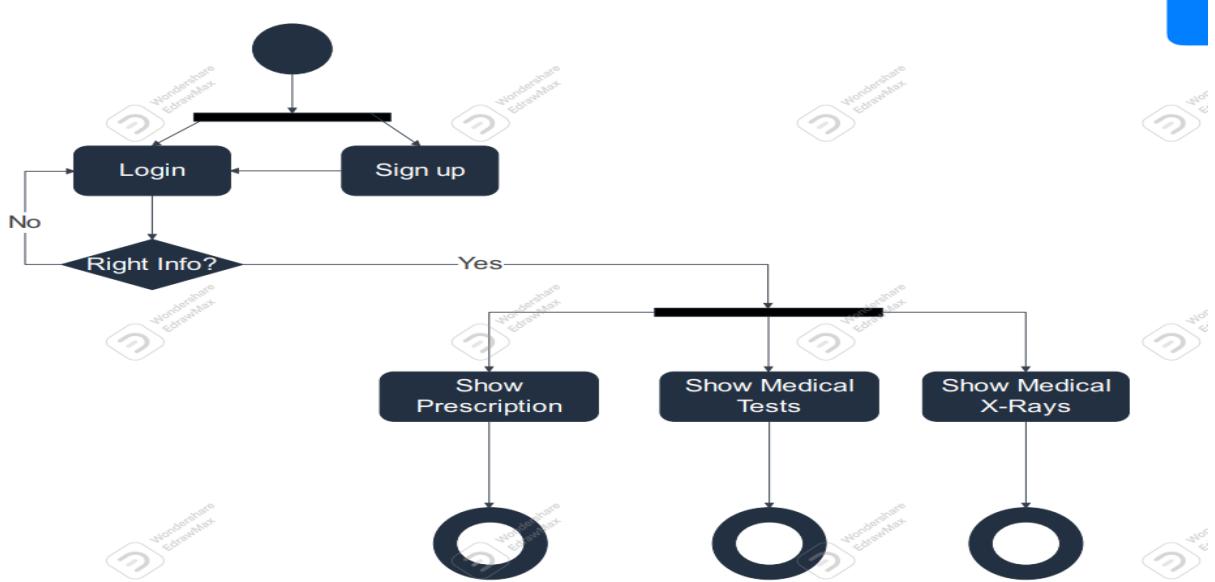
# For x-ray lab



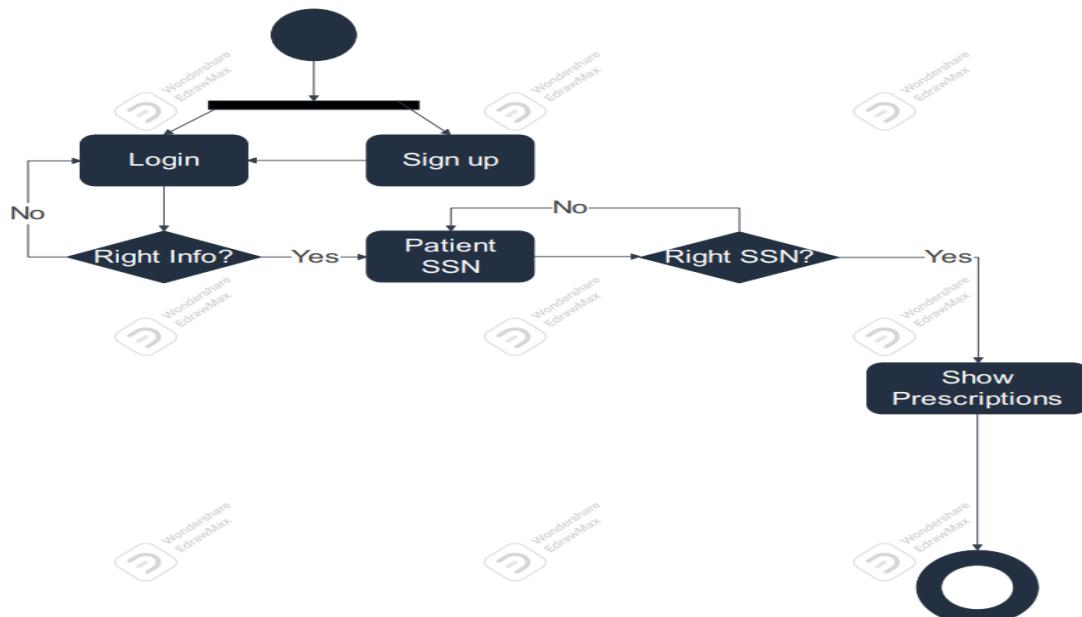
## 4.1.4. Activity For doctor



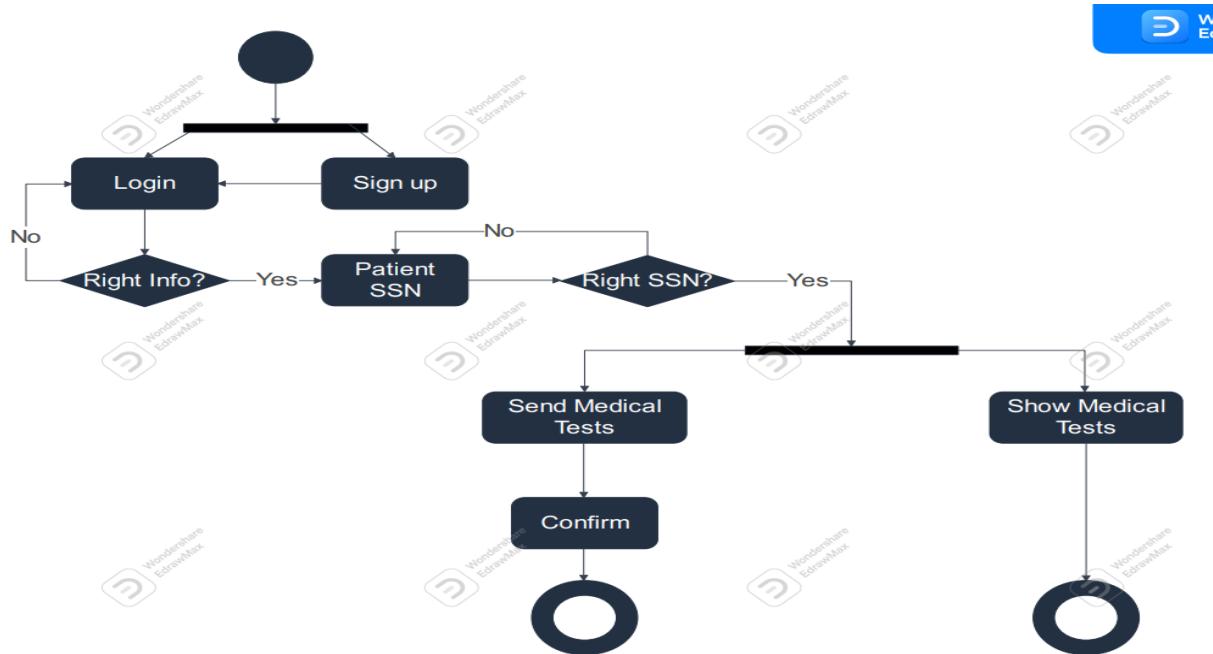
# For patient



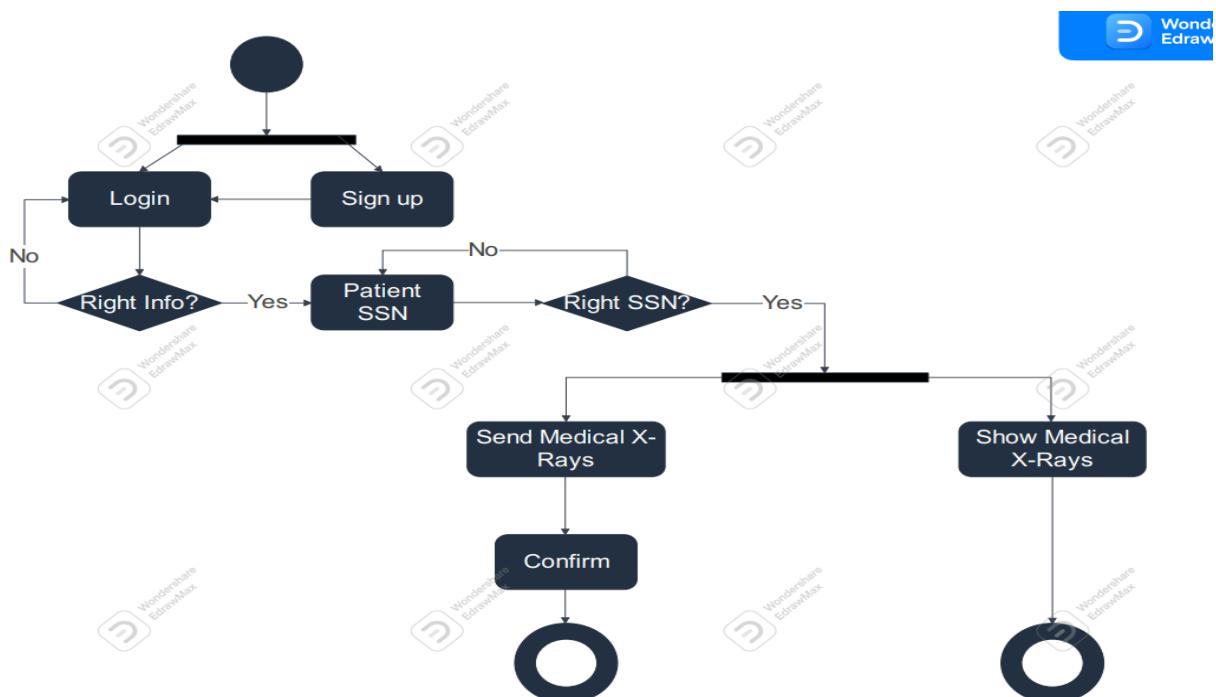
# For pharmacy



# For test lab

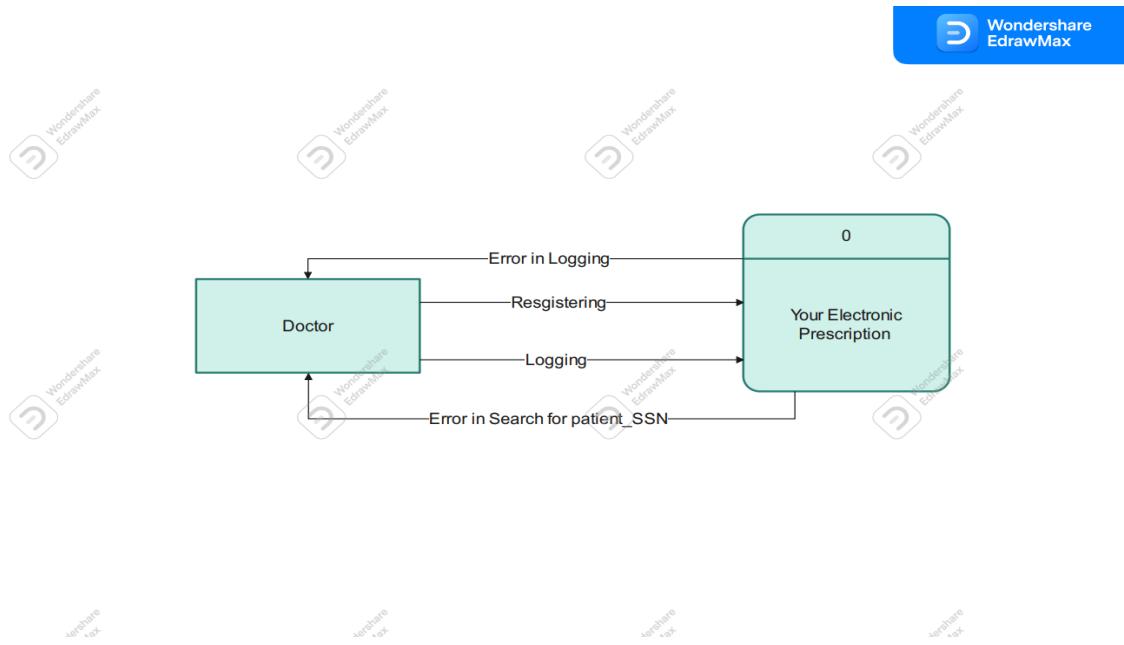


# For x-ray lab

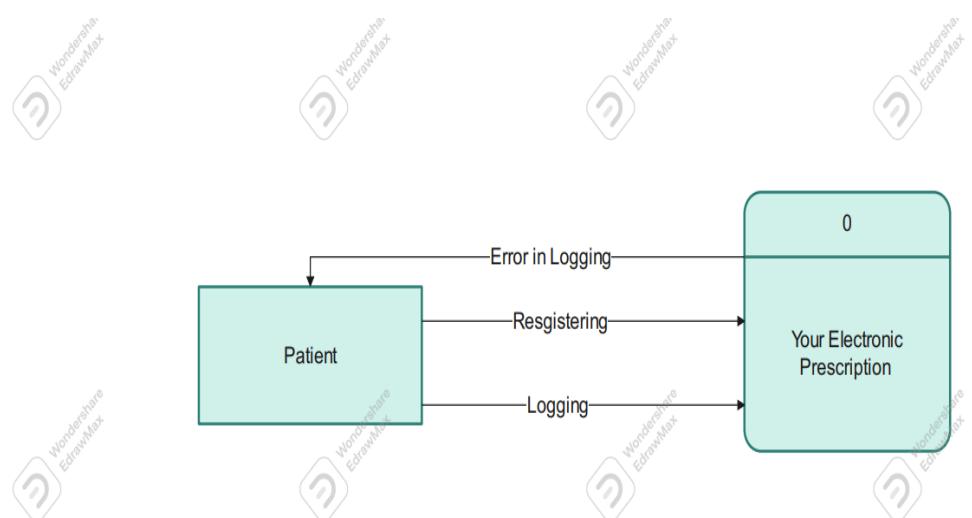


## 4.1.5. Context

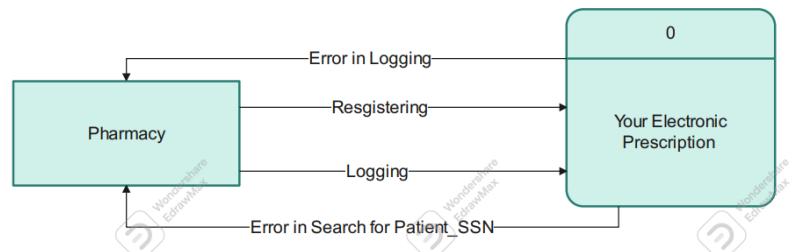
### For doctor



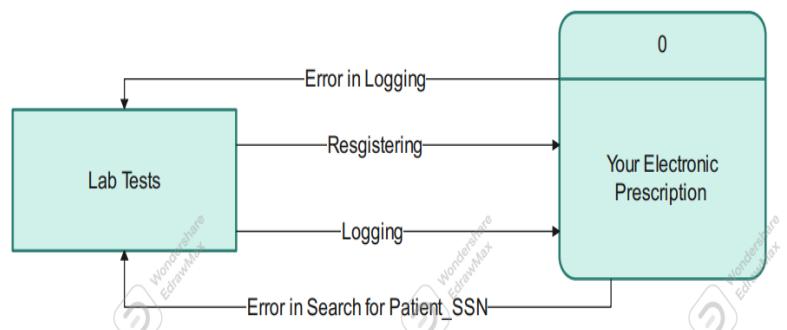
### For patient



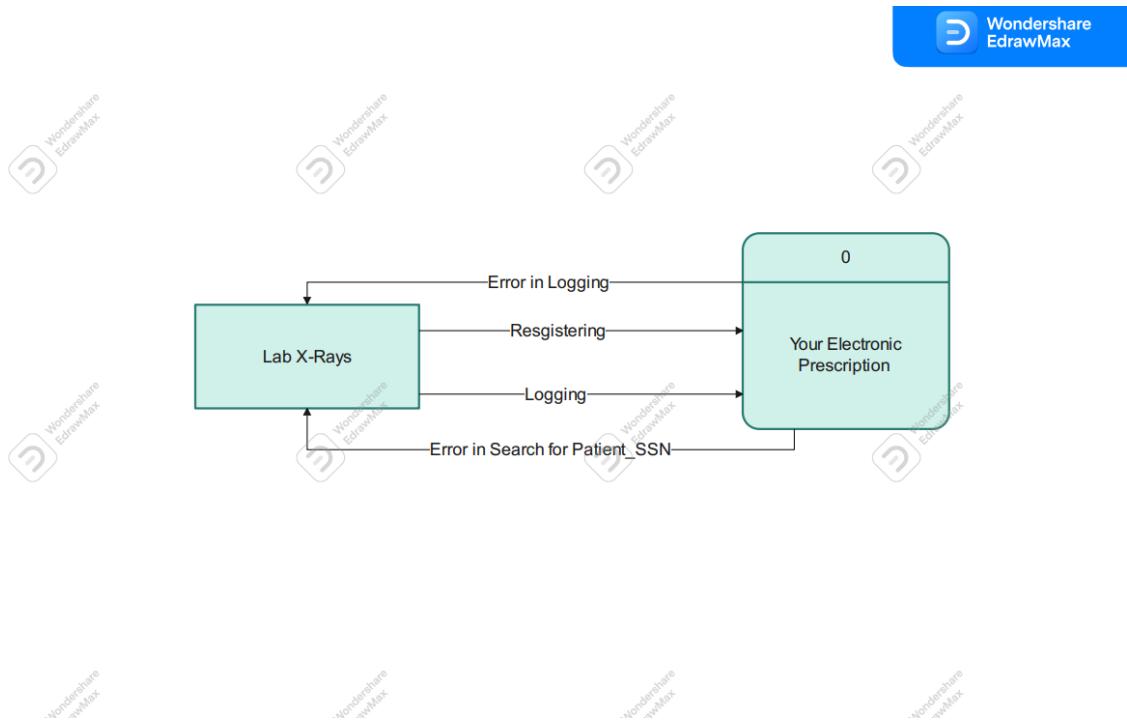
# For pharmacy



# For test lab



# For x-ray lab



## **5. Implementation:**

### **5.1.1. Development requirements**

- 7 developers:
  - .1. 3 front end
  - .2. 4 back end
- In Front end:we used :
  - .1. HTML,CSS
  - .2. Javascript,typescript
  - .3. Angular framework
- In backend:
  - .1. .net
  - .2. C#

## 5.1.2. Manual User

**Hx Wallet**

[About](#)   [Overview](#)   [Sign Up](#)   [Login](#)



**HX Wallet**  
Find your medical medicine, tests, radiology.

We will bring this to help you; we will also generate it ourselves. We will attract the very best doctors in our website to give you Your Electronic Prescription ; we will also use the newest treatments From medicines, radiology, and medical tests here ...

### About

Read to know some INFO about US



Welcome to Wallet of your Medical history

An electronic prescription with you at all times to follow up on your medications. In every follow-up with your doctor, a list of the x-rays that you have done and will do, and a list of your medical tests. We provide you with a list of doctors and pharmacies everywhere and medical laboratories to use our website in a wide field.

### OverView

Here we will explain what we can provide to you



Your doctor write your electronic prescription and you can get it from some pharmacies that connected to our App , here you can found a list of this pharmacies with their locations .

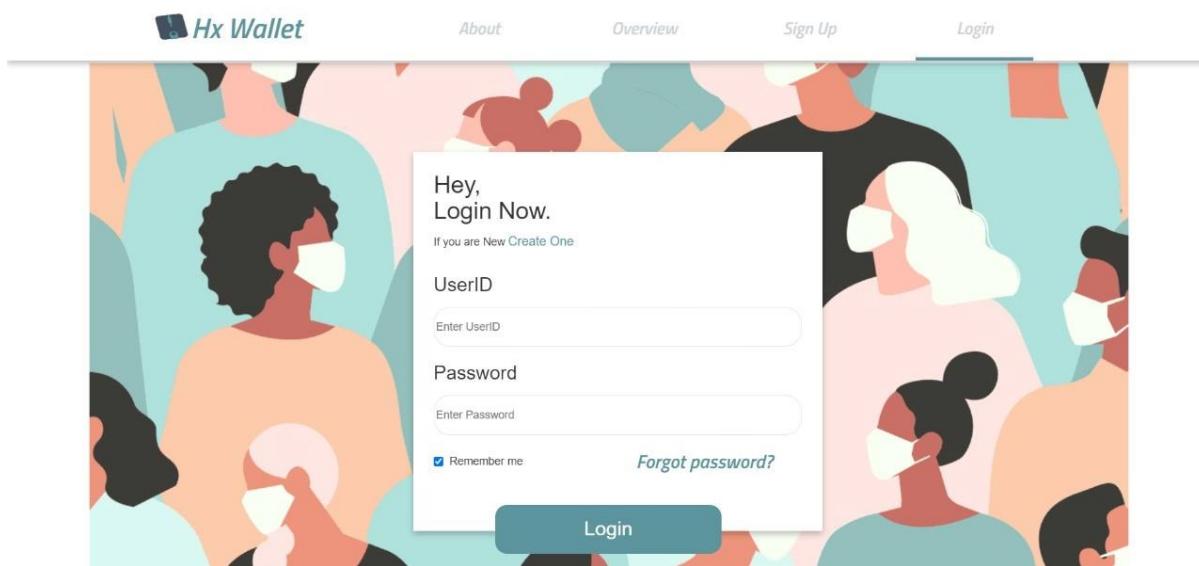
If your doctor write you for some medical tests , you can easily go to some medical laboratories linked to our App and do this tests there .

If your doctor write you for some medical X-Rays, you can easily go to some medical laboratories linked to our App and get what you want.

To see all features included in our website you must login to the site



When you click here it will lead you to this form



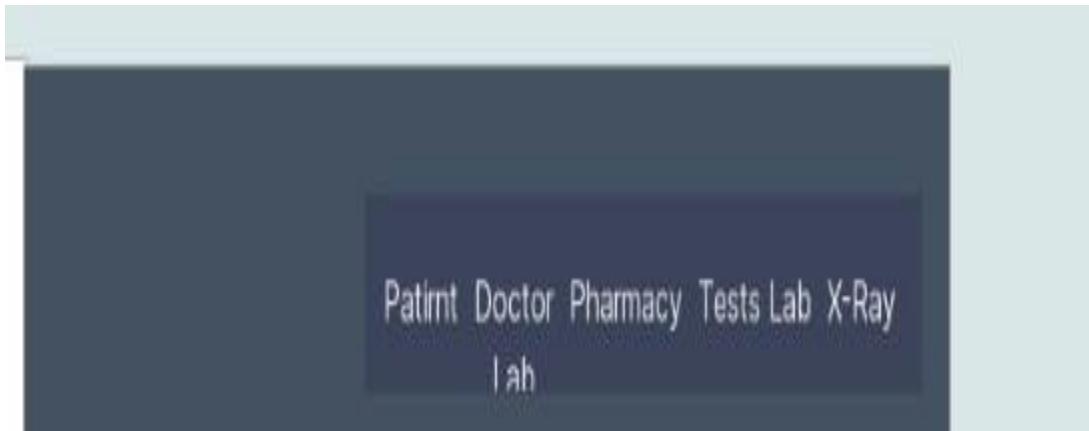
# What if you don't have an email in the website,you have to create an email



## It will direct you to signup form

A screenshot of the Hx Wallet "Sign up" form. The form is titled "Sign up" and includes a sub-instruction: "to use all features of the application". On the left side of the form, there is an illustration of a doctor wearing a stethoscope and holding a clipboard. The main form area contains fields for "Name", "ID: (National Id)", "Phone Number", "Address", "Email", "Age", "Create Password", and "Confirm Password". Each field has a placeholder text next to it. At the bottom of the form are two buttons: a teal "Sign up" button and a smaller "Already have an account? Login" link. Above the "Sign up" button, there is a small navigation bar with tabs: "Patient" (which is selected), "Doctor", "Pharmacy", "Tests Lab", and "X-Rays Lab".

## Choose what type of users is you



Each type of users have a specific form fill it and click sign up

By entering your info you are loged in and you directed to one of pages

## • Doctor page

[About](#)[Overview](#)[Sign Up](#)[Login](#)

### Actions

[show Patient Info](#)[Show Hx](#)[Add Prescription](#)[Add Medical Test](#)[Add X-Ray Test](#)

#### Medicine Name:

#### Details of medicine:

Form of medicine:

Strength:  mg Method of Intake:

#### Dosage:

Quantity:  e.g daily When:

#### Other Instructions:

[+ Add Another Dosage](#)

Note to patient e.g Medicine to be taken 30 min before meal.

#### Duration:

Start  Till:

[About](#)[Overview](#)[Sign Up](#)[Login](#)

### Actions

[show Patient Info](#)[Show Hx](#)[Add Prescription](#)[Add Medical Test](#)[Add X-Ray Test](#)[submit](#)

#### Medical X-Rays Name:

#### Other Instructions:

Note to patient .

## ● Patient page

The Patient page for medical tests displays a sidebar on the left with three icons: Prescriptions (ECG icon), Medical tests (microscope icon), and Medical X-Rays (CT scan icon). The main content area includes fields for Patient Name and Age, a section for Medical Tests Content, and a Test Details section. The Test Details section contains fields for Doc name, Test Date, and Test name, with a placeholder 'Name' in the Test name field.

**Hx Wallet**   [About](#)   [Overview](#)   [Sign Up](#)   [Login](#)

Patient Name  
Age

Medical Tests Content :

Notification :

**Test Details:**

Doc name:  
Test Date:  
Test name:

**Your Test Result:**

**Hx WALLET**  
copyright © 2022 all rights reserved.

The Patient page for prescriptions displays a sidebar on the left with three icons: Prescriptions (ECG icon), Medical tests (microscope icon), and Medical X-Rays (CT scan icon). The main content area includes fields for Patient Name and Age, a section for Prescriptions Content, and a Prescription Details section. The Prescription Details section contains fields for Doc name, Prescription Date, Medicine name, Form of Medicine, Strength, Method of Intake, Dosage, and Other Instructions. The Dosage field contains three entries: 'dose1', 'dose1', and 'dose1'. The Other Instructions field contains a single entry: 'Name'.

**Hx Wallet**   [About](#)   [Overview](#)   [Sign Up](#)   [Login](#)

Patient Name  
Age

Prescriptions Content :

Notification :

**Prescription Details:**

Doc name:  
Prescription Date:

Medicine name:   
Form of Medicine:   
Strength:   
Method of Intake:   
Dosage:   
  
  
Other Instructions:

**Hx WALLET**  
copyright © 2022 all rights reserved.

## • Pharmacy page

The screenshot displays the Hx Wallet Pharmacy interface. On the left, a sidebar lists four prescriptions, each with a "Prescription" button and the date "April 5, 2107". The main area contains a form for entering a new prescription:

**Doc name:** [Name]

**Prescription Date:** [Name]

**Medicine name:** [Name]

**Form of Medicine:** [Name]

**Strength:** [Name]

**Method of Intake:** [Name]

**Dosage:**  
dose1  
dose1  
dose1

**Other instructions:** [Name]

**Hx WALLET**  
copyright © 2022 all rights reserved.

- Test Lab page

The screenshot displays the 'Test Lab' section of the Hx Wallet application. At the top, there is a navigation bar with links for 'About', 'Overview', 'Sign Up', and 'Login'. Below the navigation, there are two separate forms side-by-side.

**Left Form:**

- Label: Doc name: (Text input field)
- Label: Test Date: (Text input field)
- Label: Test name: (Text input field)  
Sub-field: Name (Text input field)
- Label: Result  
Sub-field: Name (Text input field)

**Right Form:**

- Background: A shield-shaped graphic containing medical icons like a syringe, a bottle, and a stethoscope, surrounded by COVID-19 virus particles.
- Label: Select test: (Text input field with placeholder 'Please choose a Test')
- Label: Select result: (Text input field with placeholder 'Choose File' and 'No file chosen')
- Button: Submit (A blue button at the bottom right of the form area)

**Footer:**

Hx WALLET  
copyright © 2022 all rights reserved.

## • x-ray Lab page

Hx Wallet

Sign Up   Login

SSN:

Enter SSN

Search

Resutform

Patient ID:

X-Ray Name:

X-Ray Result:

Submit

ABOUT US

HELP

CONTACT

We are team from Assuit University,  
We aim to help you develop your  
lifestyle  
with the help of technology

We can help you  
anywhere you are,  
you can contact us.  
we are here!

Egypt , Assiut  
Info@gmail.com  
+ 20 ### #### ####  
+ 20 ### #### ####

f G in t

## 4.1.3.Screens of code

### Login code:

```
0 references
public void Configuration(IAppBuilder app)
{
    app.UseCors(CorsOptions.AllowAll);
    //use middle create tokens
    app.UseOAuthAuthorizationServer(new OAuthAuthorizationServerOptions()
    {
        TokenEndpointPath = new PathString("/login"),
        AccessTokenExpiresTimeSpan = TimeSpan.FromDays(30),
        AllowInsecureHttp = true,
        //how to create token
        Provider=new TokenCreate()
    }) ;
    app.UseOAuthBearerAuthentication(new OAuthBearerAuthenticationOptions());
    //url expire

    //configuration Object
    HttpConfiguration config = new HttpConfiguration();

    config.MapHttpAttributeRoutes();
    config.Routes.MapHttpRoute("DefaultApi", "api/{controller}/{id}",
        new { id=RouteParameter.Optional});
    app.UseWebApi(config);
    // For more information on how to configure your application, visit https://go.microsoft.com/fwlink/?LinkId=316888
}
```

```
internal class TokenCreate:OAuthAuthorizationServerProvider
{
    //#[EmailAddress]
    //public string Email { get; set; }
    0 references
    public override async Task ValidateClientAuthentication(OAuthValidateClientAuthenticationContext context)
    {
        context.Validated(); //any client id valid
    }
    //check user name pass
    0 references
    public override async Task GrantResourceOwnerCredentials
        (OAuthGrantResourceOwnerCredentialsContext context)
    {
        //Owin
        //allow access token from frontEnd
        context.OwinContext.Response.Headers.Add("Access - Control - Allow - Origin",new[] { "*"});

        //check
        UserStore<ApplicationUser> store =
            new UserStore<ApplicationUser>(new ApplicationDbContext());
        UserManager<ApplicationUser> manager =
            new UserManager<ApplicationUser>(store);
        ApplicationUser user =await manager.FindAsync(context.UserName, context.Password);
        if (user==null)
        {
            context.SetError("grant-error","User Name & Password not valid");
        }
        else
        {
            //create token
            ClaimsIdentity claim =
                new ClaimsIdentity(context.Options.AuthenticationType);
            //fields
            claim.AddClaim(new Claim(ClaimTypes.NameIdentifier, user.Id));
            claim.AddClaim(new Claim(ClaimTypes.Name, user.UserName));
            if (manager.IsInRole(user.Id, "Admin"))
            {
                claim.AddClaim(new Claim(ClaimTypes.Role, "Admin"));
            }
            context.Validated(claim);
        }
    }
}
```

## Sign up code:

- patient signup:

```

//Patient Registration
#region PatientRegistration
[Route("reg/pat")]
public async Task<IHttpActionResult> PostRegistrationPatient(PatientModel patient)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
    try
    {
        UserStore<ApplicationUser> store =
            new UserStore<ApplicationUser>(new ApplicationDbContext());
        UserManager<ApplicationUser> manager =
            new UserManager<ApplicationUser>(store);
        ApplicationUser user = new ApplicationUser();
        user.UserName = patient.Email;
        user.Email = patient.Email;
        user.PasswordHash = patient.Password;
        user.Id = patient.ID;
        user.PhoneNumber = patient.Phone;
        IdentityResult result = await manager.CreateAsync(user, patient.Password);
        if (result.Succeeded)
        {
            //Add Patient IN Database
            ApplicationDbContext appDbContext = new ApplicationDbContext();
            Patient _Patient = new Patient()
            {
                ID = patient.ID,
                Name = patient.Name,
                Email = patient.Email,
                Phone = patient.Phone,
                Age = patient.Age,
                Address = patient.Address
            };
            appDbContext.Patients.Add(_Patient);
            await appDbContext.SaveChangesAsync();
            /////////////////////////////////
            return Created("", "Register Success Patient " + user.UserName);
        }
        else
        {
            return BadRequest(result.Errors.ToList()[0]);
        }
    }
    catch (Exception ex)
    {
        return BadRequest(ex.Message);
    }
}
#endregion

```

- doctor signup:

```

//Doctor Registration
#region DoctorRegistration
[Route("reg/doc")]
0 references
public async Task<IHttpActionResult> PostRegistrationDoctor(DoctorModel doctor)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
    try
    {
        UserStore<ApplicationUser> store =
            new UserStore<ApplicationUser>(new ApplicationDbContext());
        UserManager<ApplicationUser> manager =
            new UserManager<ApplicationUser>(store);
        ApplicationUser user = new ApplicationUser();
        user.UserName = doctor.Email;
        user.PasswordHash = doctor.Password;
        user.Id = doctor.ID;
        user.PhoneNumber = doctor.Phone;
        user.Email = doctor.Email];
        IdentityResult result = await manager.CreateAsync(user, doctor.Password);
        if (result.Succeeded)
        {

            //Add Doctor IN Database
            ApplicationDbContext appDbContext = new ApplicationDbContext();
            Doctor _doctor = new Doctor()
            {
                ID = doctor.ID,
                Name = doctor.Name,
                Email = doctor.Email,
                Phone = doctor.Phone,
                Specialization = doctor.Specialization

            };
            appDbContext.Doctors.Add(_doctor);
            await appDbContext.SaveChangesAsync();
            //////////////////////////////

            return Created("", "Register Success Doctor " + user.UserName);

        }
        else
        {
            return BadRequest(result.Errors.ToList()[0]);
        }
    }
    catch (Exception ex)
    {
        return BadRequest(ex.Message);
    }
}
#endregion

```

- pharmacy signup:

```

//Pharmacy Registration
#region PharmacyRegistration
[Route("reg/phar")]
0 references
public async Task<IHttpActionResult> PostRegistrationPharmacy(PharmacyModel pharmacy)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
    try
    {
        UserStore<ApplicationUser> store =
            new UserStore<ApplicationUser>(new ApplicationDbContext());
        UserManager<ApplicationUser> manager =
            new UserManager<ApplicationUser>(store);
        ApplicationUser user = new ApplicationUser();
        user.UserName = pharmacy.Email;
        user.PasswordHash = pharmacy.Password;
        user.Id = pharmacy.ID;
        user.Email = pharmacy.Email;
        user.PhoneNumber = pharmacy.Phone;
        IdentityResult result = await manager.CreateAsync(user, pharmacy.Password);
        if (result.Succeeded)
        {
            //Add Pharmacy IN Database
            ApplicationDbContext appDbContext = new ApplicationDbContext();
            Pharmacy _Pharmacy = new Pharmacy()
            {
                ID = pharmacy.ID,
                Name = pharmacy.Name,
                Email = pharmacy.Email,
                Phone = pharmacy.Phone,
                Address = pharmacy.Address
            };
            appDbContext.Pharmacies.Add(_Pharmacy);
            await appDbContext.SaveChangesAsync();
            /////////////////////////////////
            return Created("", "Register Success Pharmacy " + user.UserName);
        }
        else
        {
            return BadRequest(result.Errors.ToList()[0]);
        }
    }
    catch (Exception ex)
    {
        return BadRequest(ex.Message);
    }
}
#endregion

```

- test lab signup:

```

//Test Lab Registration
#region TestLabRegistration
[Route("reg/tlab")]
0 references
public async Task<IHttpActionResult> PostRegistrationTestLab(TestLabModel testLab)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
    try
    {
        UserStore<ApplicationUser> store =
            new UserStore<ApplicationUser>(new ApplicationDbContext());
        UserManager<ApplicationUser> manager =
            new UserManager<ApplicationUser>(store);
        ApplicationUser user = new ApplicationUser();
        user.UserName = testLab.Email;
        user.Email = testLab.Email;
        user.PasswordHash = testLab.Password;
        user.Id = testLab.ID;
        user.PhoneNumber = testLab.Phone;
        IdentityResult result = await manager.CreateAsync(user, testLab.Password);
        if (result.Succeeded)
        {
            //Add Test Lab IN Database
            ApplicationDbContext appDbContext = new ApplicationDbContext();
            TestLab _testLab = new TestLab()
            {
                ID = testLab.ID,
                Name = testLab.Name,
                Email = testLab.Email,
                Phone = testLab.Phone,
                Address = testLab.Address
            };
            appDbContext.TestLabs.Add(_testLab);
            await appDbContext.SaveChangesAsync();
            //////////////////////////////

            return Created("", "Register Success Test Lab " + user.UserName);
        }
        else
        {
            return BadRequest(result.Errors.ToList()[0]);
        }
    }
    catch (Exception ex)
    {
        return BadRequest(ex.Message);
    }
}
#endregion
// Forget Password

```

- x-ray lab signup:

```

//X-Ray Lab Registration
#region X_RayLabRegistration
[Route("reg/xlab")]
0 references
public async Task<IHttpActionResult> PostRegistrationXRayLab(X_RayLabModel x_RayLab)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }
    try
    {
        UserStore<ApplicationUser> store =
            new UserStore<ApplicationUser>(new ApplicationDbContext());
        UserManager<ApplicationUser> manager =
            new UserManager<ApplicationUser>(store);
        ApplicationUser user = new ApplicationUser();
        user.UserName = x_RayLab.Email;
        user.Email = x_RayLab.Email;
        user.PasswordHash = x_RayLab.Password;
        user.Id = x_RayLab.ID;
        user.PhoneNumber = x_RayLab.Phone;
        IdentityResult result = await manager.CreateAsync(user, x_RayLab.Password);
        if (result.Succeeded)
        {
            //Add X-ray Lab IN Database
            ApplicationDbContext appDbContext = new ApplicationDbContext();
            X_RayLab _x_RayLab = new X_RayLab()
            {
                ID = x_RayLab.ID,
                Name = x_RayLab.Name,
                Email = x_RayLab.Email,
                Phone = x_RayLab.Phone,
                Address = x_RayLab.Address
            };
            appDbContext.X_RayLabs.Add(_x_RayLab);
            await appDbContext.SaveChangesAsync();
            /////////////////////////////////
            return Created("", "Register Success X-Ray Lab " + user.UserName);
        }
        else
        {
            return BadRequest(result.Errors.ToList()[0]);
        }
    }
    catch (Exception ex)
    {
        return BadRequest(ex.Message);
    }
}
#endregion

```

```
5 references
public class User
{
    5 references
    public string Name { set; get; }
    5 references
    public string ID { set; get; }
    5 references
    public string Phone { set; get; }
    5 references
    public string Email { set; get; }
}
```

```
3 references
public class Doctor :User
{
    1 reference
    public string Specialization { get; set; }
}
```

3 references

```
public class Patient :User
{
    |
    | 1 reference
    |     public int Age { get; set; }
    |
    | 1 reference
    |     public string Address { get; set; }
}

```

# Application DB Context:

```
41 references
public class ApplicationDbContext : IdentityDbContext//user //role //claims
{
    22 references
    public ApplicationDbContext() : base("CS"){ }
    11 references
    public virtual DbSet<Patient> Patients { get; set; }
    8 references
    public virtual DbSet<Doctor> Doctors { get; set; }
    2 references
    public virtual DbSet<Pharmacy> Pharmacies { get; set; }
    10 references
    public virtual DbSet<Prescription> Prescriptions { get; set; }
    2 references
    public virtual DbSet<XRayLab> XRayLabs { get; set; }
    0 references
    public virtual DbSet<XRay> XRays { get; set; }
    0 references
    public virtual DbSet<XRayLabPatient> XRayLabPatients { get; set; }
    2 references
    public virtual DbSet<TestLab> TestLabs { get; set; }
    0 references
    public virtual DbSet<MedicalTest> MedicalTests { get; set; }
    0 references
    public virtual DbSet<TestLabPatient> TestLabPatients { get; set; }
    7 references
    public virtual DbSet<Drug> Drugs { get; set; }
    4 references
    public virtual DbSet<DrugInPharmacy> DrugInPharmacies { get; set; }
    4 references
    public virtual DbSet<DrugInPrescription> DrugInPrescriptions { get; set; }
    6 references
    public virtual DbSet<MedicalExamination> MedicalExaminations { get; set; }
    0 references
    public virtual DbSet<Account> Accounts { get; set; }
    4 references
    public virtual DbSet<Patirnt_test> Patirnt_test { get; set; }
    4 references
    public virtual DbSet<Patient_Xray> Patient_Xray { get; set; }
    9 references
    public virtual DbSet<Notification> Notifications { get; set; }
    6 references
    public virtual DbSet<Image> Images { get; set; }
}
```



## Account Controller:

```
 0 References
public class AccountController : ApiController
{
    //Patient Registration
    [PatientRegistration]
    //Doctor Registration
    [DoctorRegistration]
    //Pharmacy Registration
    [PharmacyRegistration]
    //X-Ray Lab Registration
    [X_RayLabRegistration]
    //Test Lab Registration
    [TestLabRegistration]
}
```

**Doctor Controller:**

```
public class DoctorsController : ApiController
{
    public string connectionString = "Data Source=YOUSSEF\SQLEXPRESS;Initial Catalog=GradProject;Integrated Security=True";
    private ApplicationDbContext db = new ApplicationDbContext();
    // GET: api/Doctors
    [Route("get/docs")]
    0 references
    public IQueryable<Doctor> GetDoctors()...
    // GET: api/Doctors/5
    [ResponseType(typeof(Doctor))]
    [Route("get/doc/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetDoctor(string SSN)...
    // PUT: api/Doctors/5
    [ResponseType(typeof(void))]
    [Route("update/doc")]
    0 references
    public async Task<IHttpActionResult> PutDoctor(string SSN, Doctor doctor)...

    // POST: api/Doctors
    [Route("add/doc")]
    [ResponseType(typeof(Doctor))]
    0 references
    public async Task<IHttpActionResult> PostDoctor(Doctor doctor)...

    // DELETE: api/Doctors/5
    [ResponseType(typeof(Doctor))]
    [Route("del/doc/{SSN}")]
    0 references
    public async Task<IHttpActionResult> DeleteDoctor(string SSN)...
    protected override void Dispose(bool disposing)...

    2 references
    private bool DoctorExists(string SSN)...
}
```

## Drug Controller:

```
0 references
public class DrugsController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();
    // GET: api/Drugs
    [Route("get/drugs")]
    0 references
    public IQueryable<Drug> GetDrugs()...
        // GET: api/Drugs/5
        [ResponseType(typeof(Drug))]
        // [Route("get/drug/{id}")]
    0 references
    public async Task<IHttpActionResult> GetDrug(int id)...
        // PUT: api/Drugs/5
        [ResponseType(typeof(void))]
        [Route("update/drug")]
    0 references
    public async Task<IHttpActionResult> PutDrug(int id, Drug drug)...
        // POST: api/Drugs
        [ResponseType(typeof(Drug))]
        // [Route("add/drug")]
    0 references
    public async Task<IHttpActionResult> PostDrug(Drug drug)...
        // DELETE: api/Drugs/5
        [ResponseType(typeof(Drug))]
        [Route("del/drug")]
    0 references
    public async Task<IHttpActionResult> DeleteDrug(int id)...
    0 references
    protected override void Dispose(bool disposing)...
    2 references
    | private bool DrugExists(int id)...
}
```

## Image Controller:

```
0 references
public class ImagesController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();

    // GET: api/Images
    0 references
    public IQueryable<Image> GetImages()...

    // GET: api/Images/5
    [ResponseType(typeof(Image))]
    0 references
    public async Task<IHttpActionResult> GetImage(int id)...

    // PUT: api/Images/5
    [ResponseType(typeof(void))]
    0 references
    public async Task<IHttpActionResult> PutImage(int id, Image image)...

    // POST: api/Images
    [ResponseType(typeof(string))]
    [Consumes("multipart/form-data")]
    [System.Web.Http.Route("api/uploadImg")]
    0 references
    public async Task<IHttpActionResult> PostImage(HttpPostedFileBase postedFile)...

    // DELETE: api/Images/5
    [ResponseType(typeof(Image))]
    0 references
    public async Task<IHttpActionResult> DeleteImage(int id)...

    0 references
    protected override void Dispose(bool disposing)...

    1 reference
    private bool ImageExists(int id)...
}
```

## Notifications Controller:

## Patient- Test Controller:

```
0 references
public class NotificationsController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();
    // GET: api/Notifications
    0 references
    public IQueryable<Notification> GetNotifications()...
    // GET: api/Notifications/5
    [ResponseType(typeof(List<Notification>))]
    [Route("api/notSeenNotification/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetNotifications(String SSN)...
    [ResponseType(typeof(List<Notification>))]
    [Route("api/allNotification/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetAllNotifications(String SSN)...
    [ResponseType(typeof(int))]
    [Route("api/numberOfNotification/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetNotificationsNumber(String SSN)...
    // PUT: api/Notifications/5
    [ResponseType(typeof(void))]
    [Route("api/readAllNotifications/{SSN}")]
    0 references
    public async Task<IHttpActionResult> postreadAllNotifications(string SSN)...
    [ResponseType(typeof(void))]
    [Route("api/readOneNotification/{ID}")]
    0 references
    public async Task<IHttpActionResult> postreadOneNotification(int ID)...
    // POST: api/Notifications
    [ResponseType(typeof(void))]
    [Route("api/addNotification")]
    0 references
    public async Task<IHttpActionResult> PostNotificationTestLab(Notification notification)...
    // DELETE: api/Notifications/5
    [ResponseType(typeof(Notification))]
    0 references
    public async Task<IHttpActionResult> DeleteNotification(int id)...
}
```

```
0 references
public class Patient_testController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();

    // GET: api/Patirnt_test
    0 references
    public IQueryable<Patirnt_test> GetPatirnt_test()...[...]
}

// GET: api/Patirnt_test/5
[Route("t/{SSN}")]
0 references
public async Task<IHttpActionResult> GetPatient_tst(string SSN)...[...]
}

// PUT: api/Patirnt_test/5
[Route("add/tst")]
0 references
public async Task<IHttpActionResult> PostPatirnt_test(Patirnt_test test)...[...]
0 references
protected override void Dispose(bool disposing)
{
    if (disposing)
    {
        db.Dispose();
    }
    base.Dispose(disposing);
}
0 references
private bool Patirnt_testExists(int id)
{
    return db.Patirnt_test.Count(e => e.ID == id) > 0;
}
}
```

## Patient-X-rays Controller:

```
public class Patient_XrayController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();

    // GET: api/Patient_Xray
    0 references
    public IQueryable<Patient_Xray> GetPatient_Xray()...

    // GET: api/Patient_Xray/5
    [Route("x/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetPatient_Xray(string SSN)...
    [Route("add/xray")]
    0 references
    public async Task<IHttpActionResult> PostPatirnt_xray(Patient_Xray X_ray)...
    0 references
    protected override void Dispose(bool disposing)...
    0 references
    private bool Patient_XrayExists(int id)
    {
        return db.Patient_Xray.Count(e => e.ID == id) > 0;
    }
}
```

# Patient Controller:

```
public class PatientsController : ApiController
{
    ApplicationDbContext db = new ApplicationDbContext();
    // GET: api/Patients
    0 references
    public IQueryable<Patient> GetPatients()...
    // GET: api/Patients/5
    [ResponseType(typeof(PatientInfo))]
    [Route("Patient/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetPatient(string SSN)...

    [Route("presp/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetPrecsc(string SSN)...
    [Route("Xray/{SSN}")]
    0 references
    public async Task<IHttpActionResult>GetXray(string SSN)...

    [Route("Test/{SSN}")]
    0 references
    public async Task<IHttpActionResult> GetTest(string SSN)...
    [Route("pre/{SSN}/{idPrec}")]
    0 references
    public async Task<IHttpActionResult> Getpre(string SSN, int idPrec)...
    [Route("tpatient/{SSN}/{idtst}")]
    0 references
    public async Task<IHttpActionResult> Gettst(string SSN, int idtst)...
    [Route("xpatient/{SSN}/{idxray}")]
    0 references
    public async Task<IHttpActionResult> Getxray(string SSN, int idxray)...

    [ResponseType(typeof(Patient))]
    [Route("getPat/{SSN}")]
    0 references
    public async Task<IHttpActionResult> getPatient(string SSN)...
}
```

## Pharmacy Controller:

```
public class PharmacyController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();
    [ResponseType(typeof(void))]
    [Route("phar/BuyDrugs/{SSN}")]
    0 references
    public async Task<IHttpActionResult> postBuyDrugs(string SSN,BuyDrug buyDrug)...
    [ResponseType(typeof(List<Prescription>))]
    [Route("phar>ShowPatientPres/{SSN}")]
    0 references
    public async Task<IHttpActionResult> getPatientPrescriptions(string SSN)...
    [ResponseType(typeof(List<DrugInfo>))]
    [Route("phar>ShowDrugs/{SSN}/{ID}")]
    0 references
    public async Task<IHttpActionResult> getShowPrescription(string SSN,int ID)...
    [ResponseType(typeof(double))]
    [Route("phar>VentDrugs/{SSN}/{ID}")]
    0 references
    public async Task<IHttpActionResult> postVentPrescription(string SSN,int ID,List<DrugInfo> drugInfos)...
    0 references
    protected override void Dispose(bool disposing)...
    0 references
    private bool PrescriptionExists(int id)...
}
```

# Prescription Controller:

```
public class PrescriptionsController : ApiController
{
    private ApplicationDbContext db = new ApplicationDbContext();
    // GET: api/Prescriptions
    [Route("get/pres")]
    0 references
    public IQueryable<Prescription> GetPrescriptions()...

    // GET: api/Prescriptions/5
    [ResponseType(typeof(Prescription))]
    // [Route("get/pre/{id}")]
    // [Route("api/pre/{id}", Name = "GetPreById")]
    0 references
    public async Task<IHttpActionResult> GetPrescription(int id)...

    // PUT: api/Prescriptions/5
    [ResponseType(typeof(void))]
    [Route("update/pre")]
    0 references
    public async Task<IHttpActionResult> PutPrescription(int id, Prescription prescription)...

    // POST: api/Prescriptions
    [ResponseType(typeof(Prescription))]
    // [Route("add/pre")]
    0 references
    public async Task<IHttpActionResult> PostPrescription(PrescriptionModel prescriptionModel)...

    // DELETE: api/Prescriptions/5
    [ResponseType(typeof(Prescription))]
    [Route("del/pre/{id}")]
    0 references
    public async Task<IHttpActionResult> DeletePrescription(int id)...
    0 references
    protected override void Dispose(bool disposing)...
    2 references
    private bool PrescriptionExists(int id)...
}
```

### **5.1.3. References**

in front end:

- ITI front end training course
- Google search

In back end:

- ITI .net training course
- Google search
- Asp.net web api

## **6. Conclusion and Future Improvement:**

### **6.1.1. Conclusion:**

we did website that make a contact between five stackholders with each other they are (Doctor, patient, pharmacy, x-rays lab, test-lab) and it focus on the prescription but the prescription is electronic because this idea make all the

patient history saved easier and no any treatment details are loss.

### **6.1.2. Future Improvement:**

- 1-Stay Focused on the Main Goal.
- 2-Improve Project Planning and Quality.
- 3-increase security.
- 4-Use Intuitive Time and Expense Technology.
- 5-Increase productivity.
- 6-Minimize risks and high return on investement.
- 7-Add financials to the project status report.
- 8-Send out meeting agendas in advance.
- 9-Revise project financials weekly.
- 10-Revisit resource forecasts weekly.

