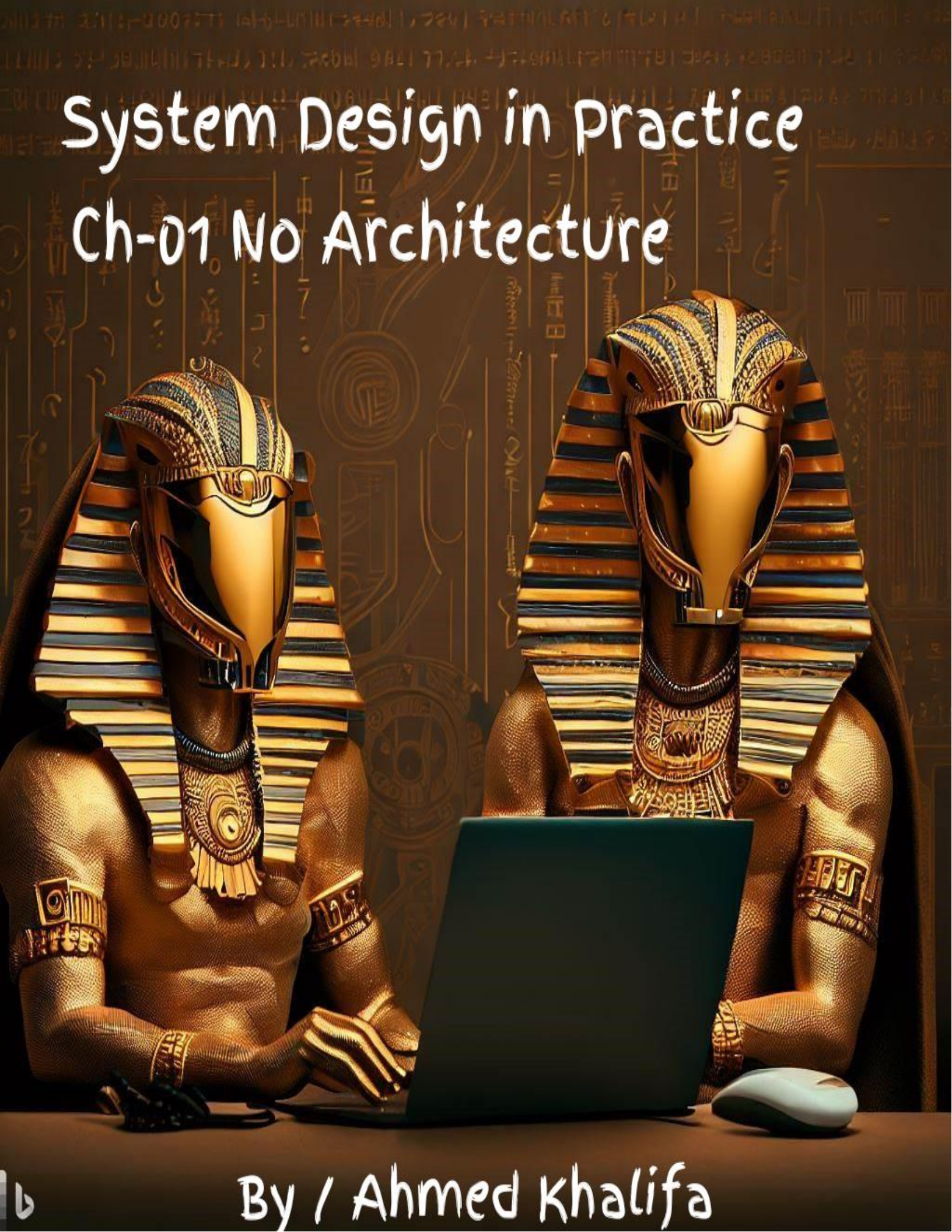
System Design in Practice

Ch-01 No Architecture



By / Ahmed Khalifa

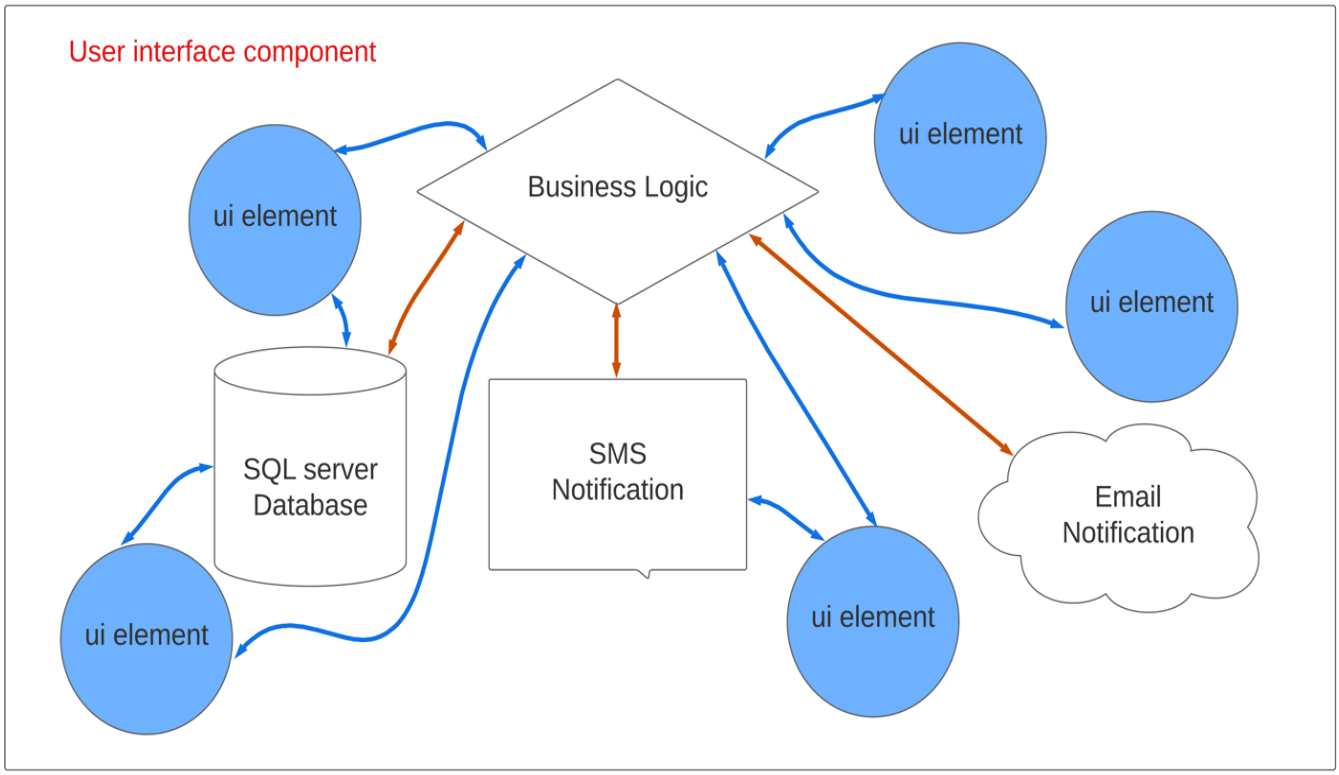
Index

1.Koshary architecture :

- Koshary architecture is just a joke like spaghetti code but from the architecture perspective which mean there are no actually architecture pattern exist or the legacy project.
- Before studying system design and architecture patterns we need first to know what are the problems that will be in our system when there are no architecture pattern used ?

Koshary is a delicious Egyptian food that contains a mixture of different food elements in single dish.

Koshary architecture pattern is refer to the architecture pattern which single component (usually the user interface component) contain the implementation of all other components in the system like :
business logic ,
database ,
notifications
and .. etc.

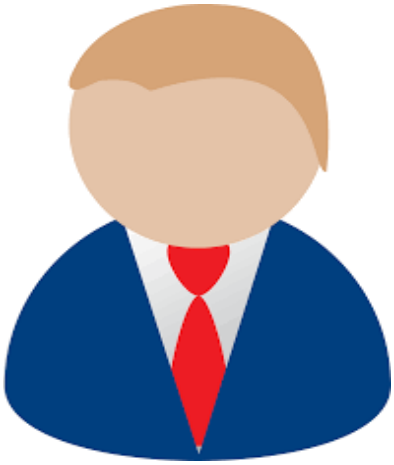


Koshary architecture

2. User story : Add new employee by manager

To better understanding, let's see that with real user story and actual project (employee management system) with code sample :

2.1 User story : agile sample



As a manager,

I want to add new employee to the system,

So I can save important information about the employee like
- personal information : name , birth date
- contact info : email , phone number
- address : country , city , street
- skills

2.2 User story inputs :

Data	Data type
Employee Name	string
Employee Birthdate	Datetime
Email	string
Phone number	string
Country	string
City	string
Street and Building Number	string
Skills	List

2.3 User story business logic :

- Logic Workflow :

User	System
Manager opens screen called “Add new Employee”.	System display “Add new Employee” screen.
Manager enters the information of the employee on the UI elements and inputs.	-
Manager click on button called “Add new Employee”	System will : 1. if employee information is valid , then system will save the new employee and display success message 2. if employee information is not valid , then system will not save the new employee and display errors message

- Business rules:

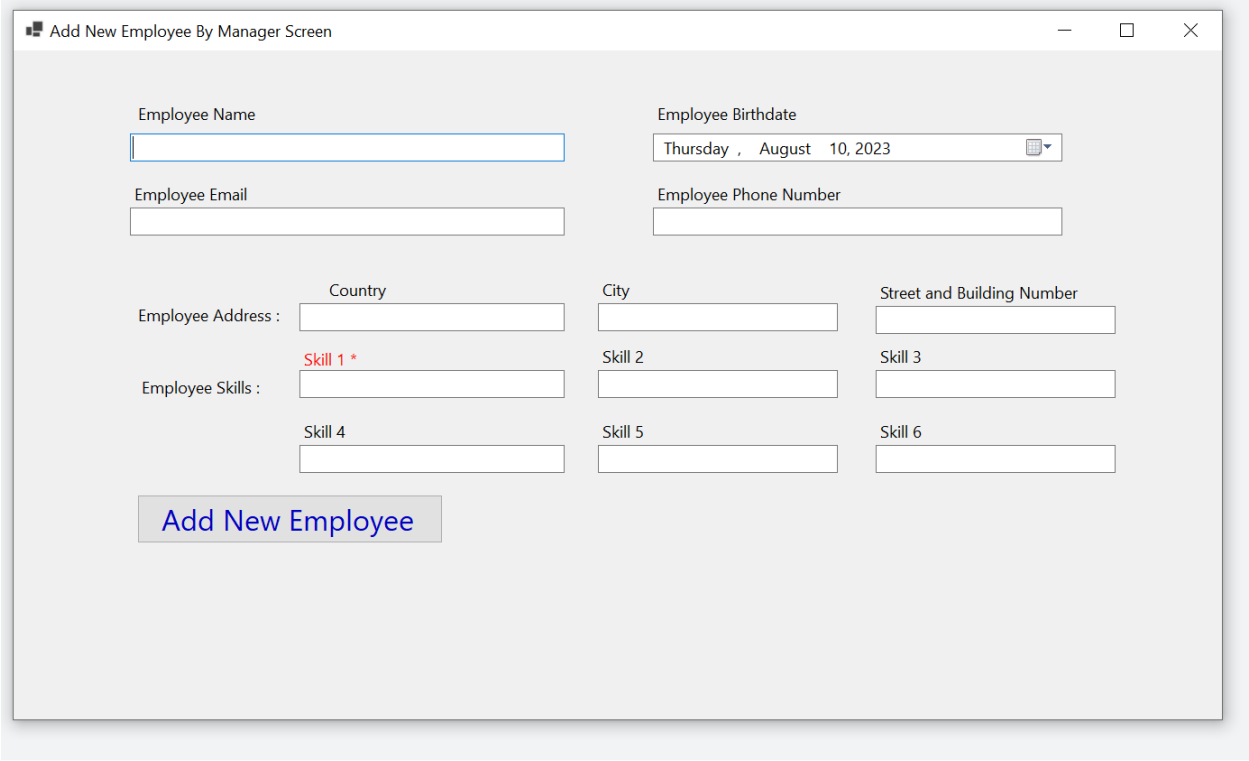
1. Employee age should be equal or greater than 21
2. Employee email must has correct format like test@test.com but the system should not accept other incorrect formats like test@test , test_test.com.
3. Employee phone number is required and must be unique , system can not have two employees with the same phone number.
4. Employee must has at min 1 skill.

2.4 User story outputs :

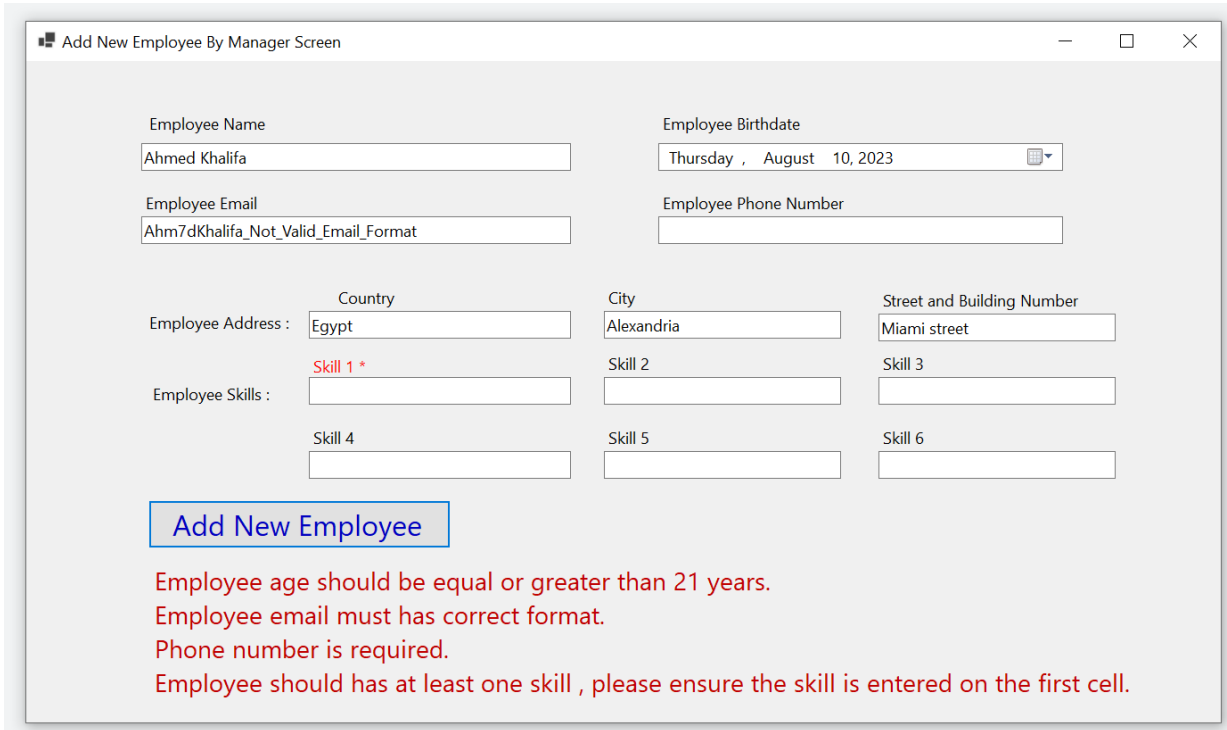
Case	Result
Case 1 : input by manager apply the business rules	1. Save new employee on database 2. Send Email Notification to Admin 3. Display success message on the screen for the manager
Case 2 : Input by manager does not apply the business rules	1. Display errors message on the screen for the manager

2.5 User story UI/UX :

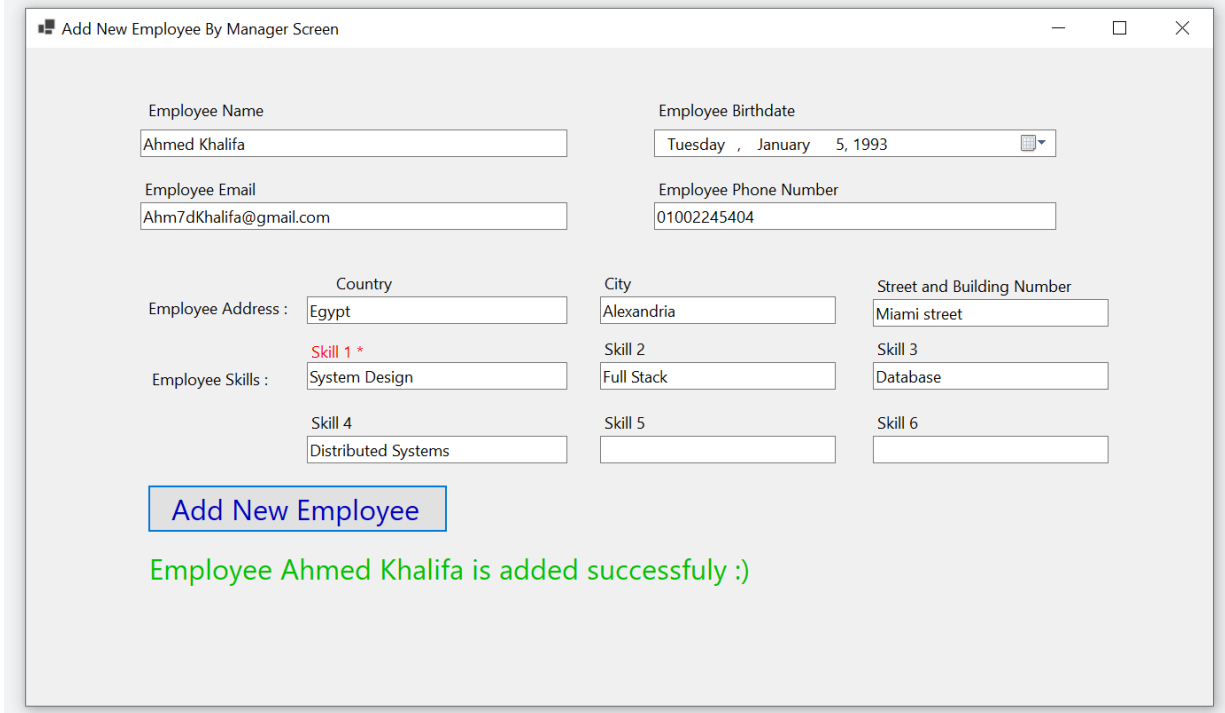
- Default screen :



- Screen with errors message :



- Screen with success message :



2.6 User story code sample :

```
9   namespace Koshary_Architecture
10  {
        3 references | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
11      public partial class AddNewEmployeeByManagerForm : Form
12      {
13          Properties
            1 reference | Ahmed Khalifa, 1 day ago | 1 author, 1 change
18          public AddNewEmployeeByManagerForm()
19          {
20              InitializeComponent();
21          }
            1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
22          private void AddNewEmployeeButton_Click(object sender, EventArgs e)
23          {
24              InitUiConfiguration();
25
26              CheckIfEmployeeAgeIsEqualOrGreaterThan21Years();
27              CheckIfEmailHasCorrectFormat();
28              CheckIfEmployeePhoneNumberIsRequiredAndUnique();
29              CheckIfEmployeeHasAtLeastOneSkill();
30
31              if (Errors.Count > 0)
32              {
33                  DisplayErrorMessage();
34              }
35              else
36              {
37                  CreateNewEmployee();
38                  SaveNewEmployeeOnDatabase();
39                  SendEmailNotificationToAdmin();
40                  DisplaySuccessMessage();
41              }
42          }
```

```
1 reference | Ahmed Khalifa, 2 days ago | 1 author, 1 change
44  private void InitUiConfiguration()
45  {
46      Errors = new List<string>();
47
48      NewEmployee = new Employee();
49
50      SuccessMessageLabel.Text = string.Empty;
51      SuccessMessageLabel.Visible = false;
52      ErrorsMessageLabel.Text = string.Empty;
53      ErrorsMessageLabel.Visible = false;
54  }
55

1 reference | Ahmed Khalifa, 2 days ago | 1 author, 2 changes
56  private void CheckIfEmployeeAgeIsEqualOrGreaterThan21Years()
57  {
58      DateTime today = DateTime.Now;
59      var age = today.Year - EmployeeBirthDatePickerBox.Value.Year;
60      if (age < 21)
61      {
62          Errors.Add("Employee age should be equal or greater than 21 years.");
63      }
64  }
65

1 reference | Ahmed Khalifa, 2 days ago | 1 author, 1 change
66  private void CheckIfEmailHasCorrectFormat()
67  {
68      bool isEmail = Regex.IsMatch(EmployeeEmailTextBox.Text,
69                                   @"\A(?:[a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)
70                                   RegexOptions.IgnoreCase);
71
72      if (!isEmail)
73      {
74          Errors.Add("Employee email must has correct format.");
75      }
76
77  }
78

1 reference | Ahmed Khalifa, 22 hours ago | 1 author, 1 change
79  private void CheckIfEmployeePhoneNumberIsRequiredAndUnique()
80  {
81      if (string.IsNullOrEmpty(EmployeePhoneNumberTextBox.Text))
82      {
83          Errors.Add("Phone number is required. ");
84      }
85
86      bool isPhoneNumberExist = SqlServerDatabaseContext
87                    .Employees
88                    .Where(e => e.PhoneNumber == EmployeePhoneNumberTextBox.Text)
89                    .Any();
90      if (isPhoneNumberExist)
91      {
92          Errors.Add("Phone number is already exist , please enter unique phone number. ");
93      }
94  }
95
```

```
96
                1 reference | Ahmed Khalifa, 2 days ago | 1 author, 2 changes
97              private void CheckIfEmployeeHasAtLeastOneSkill()
98              {
99                  if (string.IsNullOrWhiteSpace(EmployeeSkillTextBox_1.Text))
100                 {
101                     Errors.Add("Employee should has at least one skill , " +
102                                "please ensure the skill is entered on the first cell. ");
103                 }
104             }
105
                1 reference | Ahmed Khalifa, 2 days ago | 1 author, 1 change
106             private void CreateNewEmployee()
107             {
108                 NewEmployee = new Employee();
109                 NewEmployee.Name = EmployeeNameTextBox.Text;
110                 NewEmployee.Birthdate = EmployeeBirthDatePickerBox.Value;
111                 NewEmployee.Email = EmployeeEmailTextBox.Text;
112                 NewEmployee.PhoneNumber = EmployeePhoneNumberTextBox.Text;
113                 NewEmployee.Country = CountryTextBox.Text;
114                 NewEmployee.City = CityTextBox.Text;
115                 NewEmployee.StreetAndBuildingNumber = EmployeeStreetAndBuildingNumberTextBox.Text;
116
117                 List<EmployeeSkill> employeeSkills = new List<EmployeeSkill>();
118                 employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_1.Text });
119                 if (!string.IsNullOrEmpty(EmployeeSkillTextBox_2.Text))
120                     employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_2.Text });
121                 if (!string.IsNullOrEmpty(EmployeeSkillTextBox_3.Text))
122                     employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_3.Text });
123                 if (!string.IsNullOrEmpty(EmployeeSkillTextBox_4.Text))
124                     employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_4.Text });
125                 if (!string.IsNullOrEmpty(EmployeeSkillTextBox_5.Text))
126                     employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_5.Text });
127                 if (!string.IsNullOrEmpty(EmployeeSkillTextBox_6.Text))
128                     employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_6.Text });
129
130                 NewEmployee.EmployeeSkills = employeeSkills;
131             }
132
```

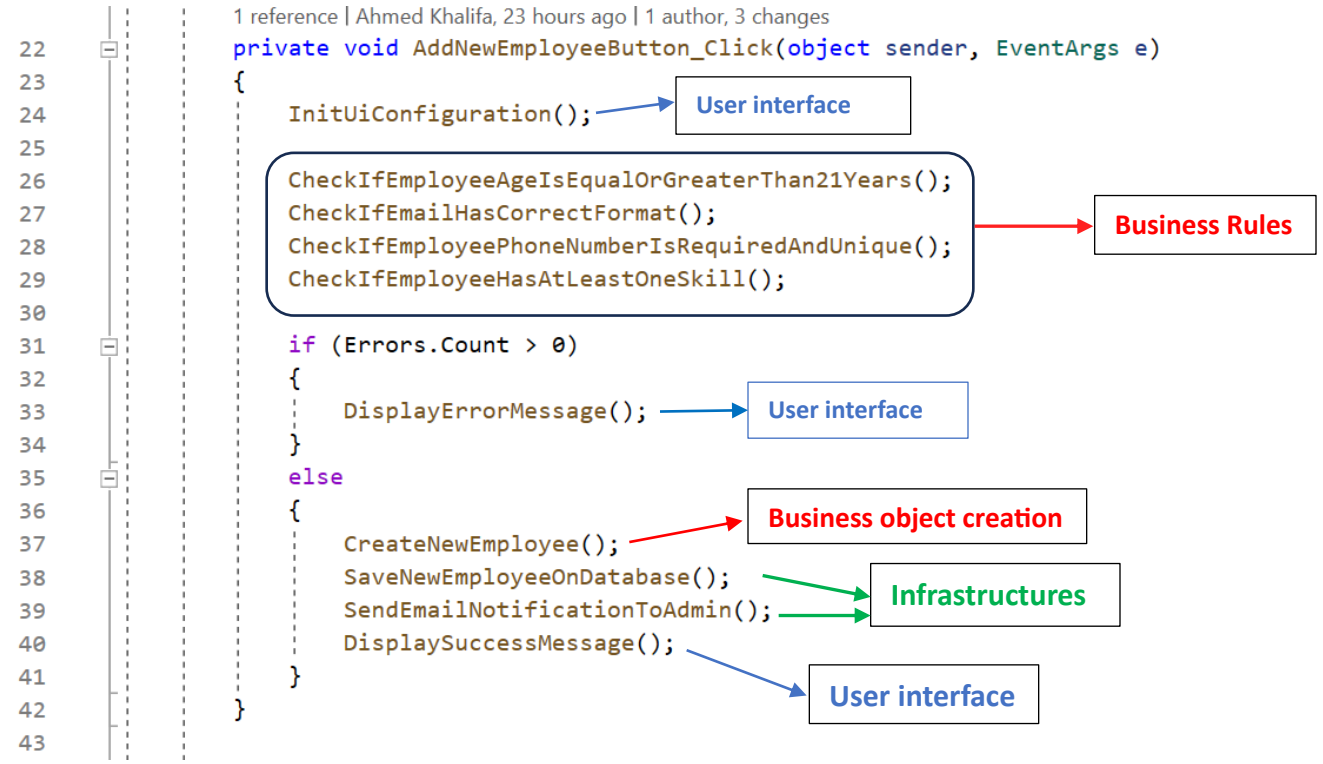
```
                1 reference | Ahmed Khalifa, 2 days ago | 1 author, 1 change
133             private void SaveNewEmployeeOnDatabase()
134             {
135                 SqlServerDatabaseContext.Employees.Add(NewEmployee);
136                 SqlServerDatabaseContext.SaveChanges();
137             }
138
                1 reference | Ahmed Khalifa, 2 days ago | 1 author, 1 change
139             private void SendEmailNotificationToAdmin()
140             {
141                 Console.WriteLine("Email is sent ...");
142                 Console.WriteLine("you can uncomment the code below to send actual email ");
143                 /*
144                 https://www.c-sharpcorner.com/article/sending-email-using-c-sharp/
145                 MailMessage message = new MailMessage();
146                 SmtpClient smtp = new SmtpClient();
147                 message.From = new MailAddress("FromMailAddress");
148                 message.To.Add(new MailAddress("ToMailAddress"));
149                 message.Subject = "Test";
150                 message.IsBodyHtml = true; //to make message body as html
151                 message.Body = htmlString;
152                 smtp.Port = 587;
153                 smtp.Host = "smtp.gmail.com"; //for gmail host
154                 smtp.EnableSsl = true;
155                 smtp.UseDefaultCredentials = false;
156                 smtp.Credentials = new NetworkCredential("FromMailAddress", "password");
157                 smtp.DeliveryMethod = SmtpDeliveryMethod.Network;
158                 smtp.Send(message);
159                 */
160             }
161
162
                1 reference | Ahmed Khalifa, 22 hours ago | 1 author, 2 changes
163             private void DisplaySuccessMessage()
164             {
165                 SuccessMessageLabel.Text = "Employee " + NewEmployee.Name + " is added successfuly :)";
166                 SuccessMessageLabel.Visible = true;
167             }
168
                1 reference | Ahmed Khalifa, 2 days ago | 1 author, 1 change
169             private void DisplayErrorMessage()
170             {
171                 string errorMessage = string.Empty;
172                 foreach (var error in Errors)
173                 {
174                     errorMessage += error + "\r\n";
175                 }
176                 ErrorsMessageLabel.Text = errorMessage;
177                 ErrorsMessageLabel.Visible = true;
178             }
179
180         }
181     }
```

3. Problems of koshary architecture :

This code may be apply some of the clean code principles like good naming , small functions but from the architecture perspective has a lot of problems.

First problem (No Separation Of Concern) :

The user interface component : **AddEmployeeByManagerForm** contain **the actual implementation** of all other components like business rules , database and email.



The current architecture is not layered or has any modules to separate responsibilities from each other , just one layer (user interface) contain the implementation of all things.

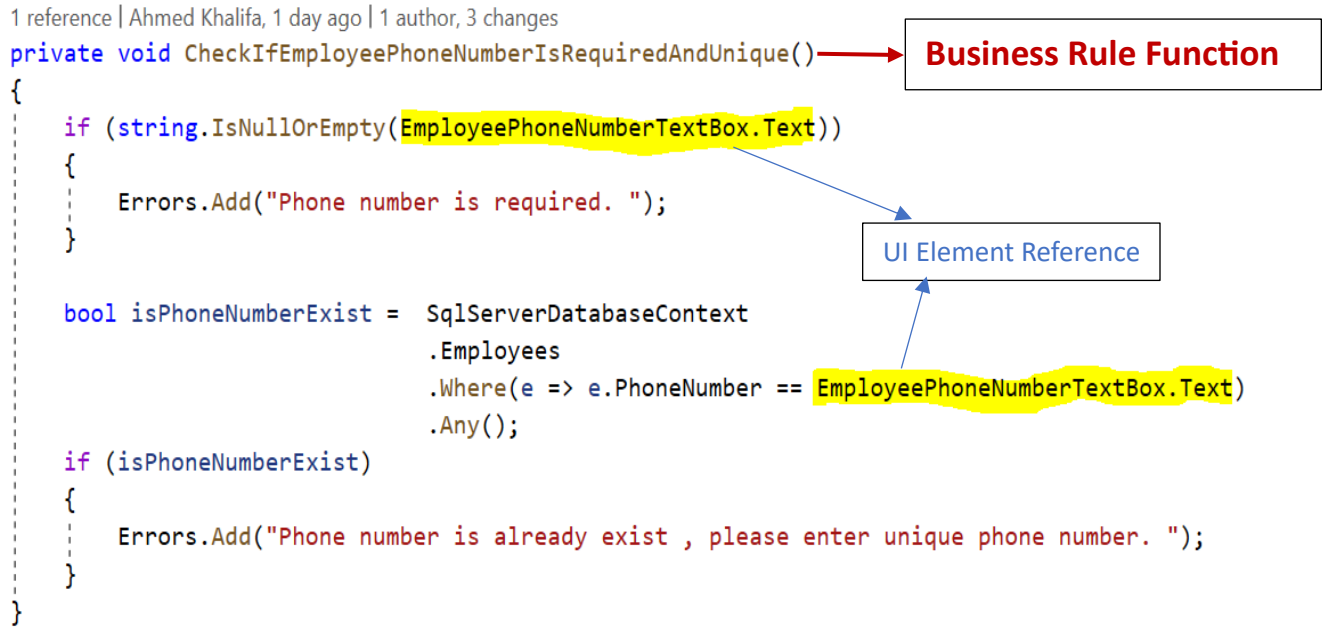
Now if the company need to change the UI desktop framework form c# windows form that run only in the windows os to other UI desktop frameworks like .NET MAUI which is a cross platform that run on windows , mac , android and ios , this means the developers will not only change the code base of UI elements or framework , but they need to completely refactor the project from scratch.

Second problem :

business logic is very coupled and depended on the user interface

Any change on the user interface code will effect on the business logic

For example :



This business function contain reference to the ui element , this cause the business function will be coupled on the ui element.

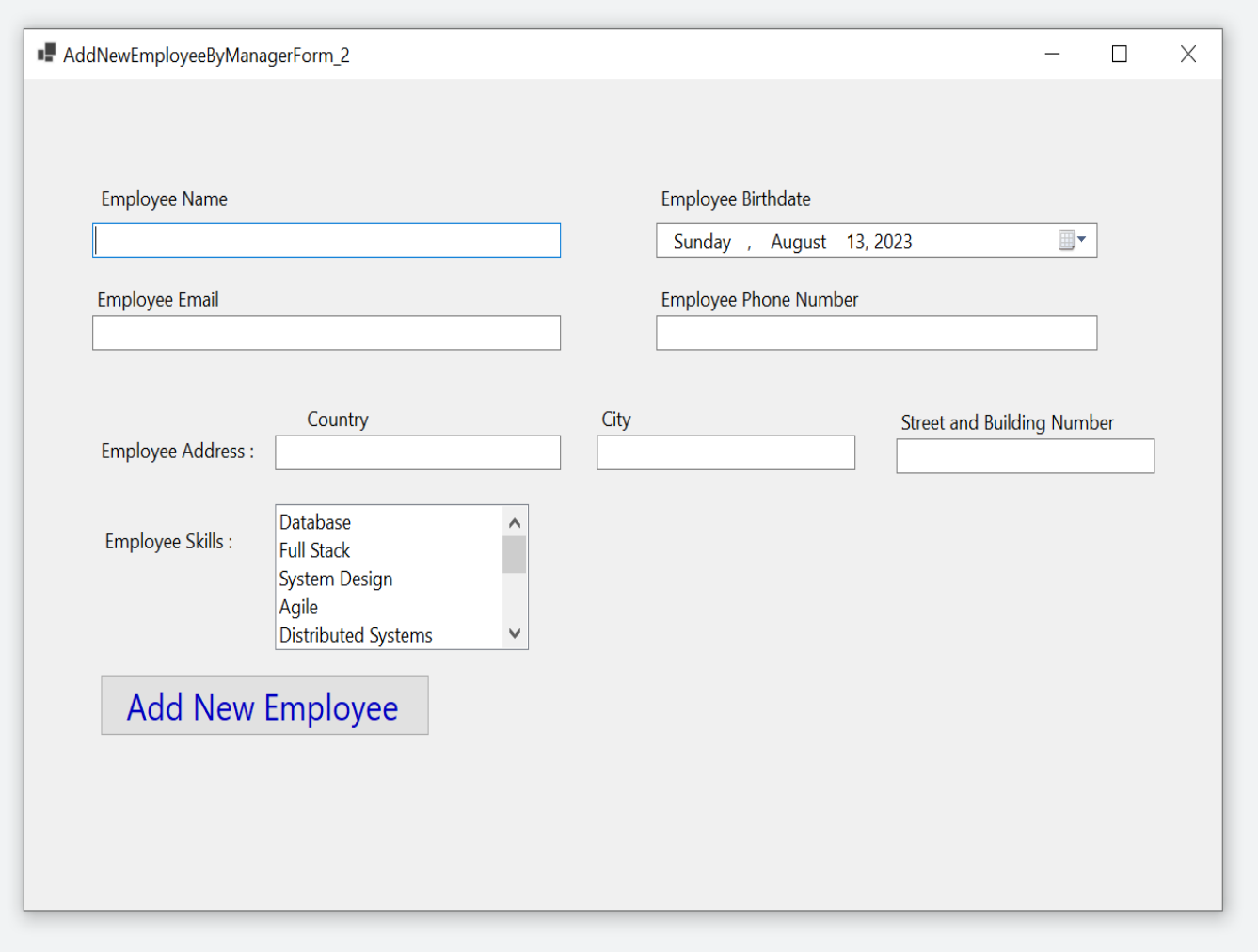
for example :

changing the ui element name from EmployeePhoneNumberTextBox to PhoneNumberTextBox will have effect side on this function and the developer need to revisit and fix this function :

```
1 reference | Ahmed Khalifa, 1 day ago | 1 author, 3 changes
private void CheckIfEmployeePhoneNumberIsRequiredAndUnique()
{
    if (string.IsNullOrEmpty(PhoneNumberTextBox.Text))
    {
        Errors.Add("Phone number is required. ");
    }

    bool isPhoneNumberExist =  SqlServerDatabaseContext
                               .Employees
                               .Where(e => e.PhoneNumber == PhoneNumberTextBox.Text)
                               .Any();
    if (isPhoneNumberExist)
    {
        Errors.Add("Phone number is already exist , please enter unique phone number. ");
    }
}
```

You can think this is just a small fix , but in the real life and projects which have complex business logic and complex user interface the changing on the ui can have a very risk and dangerous effect on the business logic.

let’s see more complex example : change the skills text boxes to drop down menu that has multi select options :



This ui change will have effect side on two business functions :

1. CheckIfEmployeeHasAtLeastOneSkill()
2. CreateNewEmployee()

Old code :

```
1 reference | Ahmed Khalifa, 4 days ago | 1 author, 2 changes
private void CheckIfEmployeeHasAtLeastOneSkill()
{
    if (string.IsNullOrWhiteSpace(EmployeeSkillTextBox_1.Text))
    {
        Errors.Add("Employee should has at least one skill , " +
                   "please ensure the skill is entered on the first cell. ");
    }
}
```

```
1 reference | Ahmed Khalifa, 4 days ago | 1 author, 1 change
107     private void CreateNewEmployee()
108     {
109         NewEmployee = new Employee();
110         NewEmployee.Name = EmployeeNameTextBox.Text;
111         NewEmployee.Birthdate = EmployeeBirthDatePickerBox.Value;
112         NewEmployee.Email = EmployeeEmailTextBox.Text;
113         NewEmployee.PhoneNumber = EmployeePhoneNumberTextBox.Text;
114         NewEmployee.Country = CountryTextBox.Text;
115         NewEmployee.City = CityTextBox.Text;
116         NewEmployee.StreetAndBuildingNumber = EmployeeStreetAndBuildingNumberTextBox.Text;
117 
118         List<EmployeeSkill> employeeSkills = new List<EmployeeSkill>();
119         employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_1.Text });
120         if (!string.IsNullOrEmpty(EmployeeSkillTextBox_2.Text))
121             employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_2.Text });
122         if (!string.IsNullOrEmpty(EmployeeSkillTextBox_3.Text))
123             employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_3.Text });
124         if (!string.IsNullOrEmpty(EmployeeSkillTextBox_4.Text))
125             employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_4.Text });
126         if (!string.IsNullOrEmpty(EmployeeSkillTextBox_5.Text))
127             employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_5.Text });
128         if (!string.IsNullOrEmpty(EmployeeSkillTextBox_6.Text))
129             employeeSkills.Add(new EmployeeSkill { SkillName = EmployeeSkillTextBox_6.Text });
130 
131         NewEmployee.EmployeeSkills = employeeSkills;
132     }
```

New code :

```
1 reference | 0 changes | 0 authors, 0 changes
private void CheckIfEmployeeHasAtLeastOneSkill()
{
    if (SkillsDropDown.SelectedItems.Count <= 0)
    {
        Errors.Add("Employee should has at least one skill. ");

    }
}
```

```
1 reference | 0 changes | 0 authors, 0 changes
private void CreateNewEmployee()
{
    NewEmployee = new Employee();
    NewEmployee.Name = EmployeeNameTextBox.Text;
    NewEmployee.Birthdate = EmployeeBirthDatePickerBox.Value;
    NewEmployee.Email = EmployeeEmailTextBox.Text;
    NewEmployee.PhoneNumber = EmployeePhoneNumberTextBox.Text;
    NewEmployee.Country = CountryTextBox.Text;
    NewEmployee.City = CityTextBox.Text;
    NewEmployee.StreetAndBuildingNumber = EmployeeStreetAndBuildingNumberTextBox.Text;

    List<EmployeeSkill> employeeSkills = new List<EmployeeSkill>();
    foreach (var selectedSkill in SkillsDropDown.SelectedItems)
    {
        employeeSkills.Add(new EmployeeSkill { SkillName = selectedSkill.ToString() });
    }
    NewEmployee.EmployeeSkills = employeeSkills;
}
```

In the real projects with complex business logic and user interface , changing on the ui can cause a lot of conflicts and bugs when the business logic is very coupled and depended on the user interface , some developers can change the actual business logic by a mistake.

Third problem :

business logic is very coupled and depended on the infrastructure components like database :

infrastructure is refer to :

1.the physical components in the system like databases or file system

2.external services like sms , emails , notifications and payments , this services are depended on external systems and usually the application interact with them through third party api , web services , or library or framework that the application setup them .

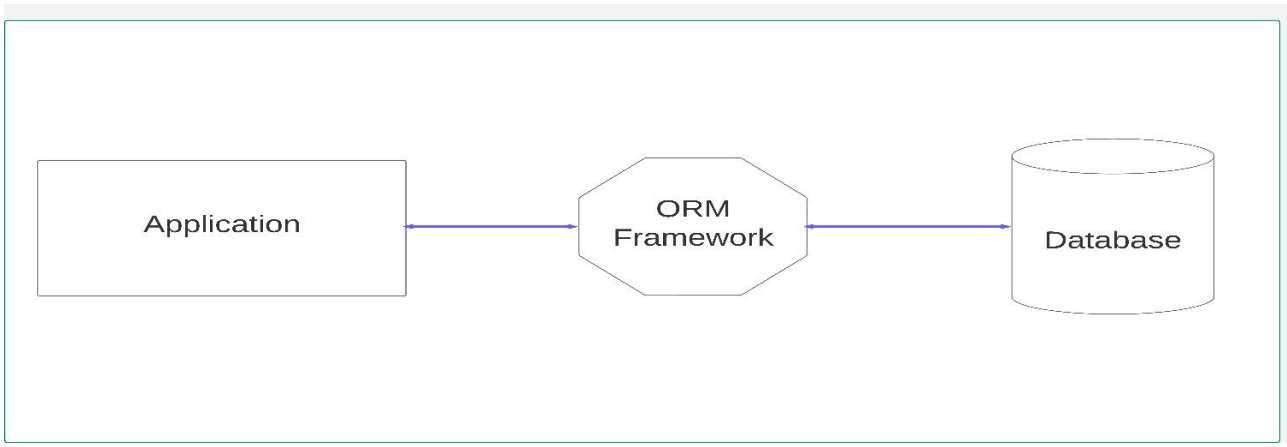
We will take database as infrastructure example : Applications usually need an orm framework to connect to the database so they can write or read data from the database.

the orm frameworks like entity framework core , dapper , ado.net are responsible :

1- managing this communications

2- Read and write operations

2- mapping between database rows and application in-memory objects.

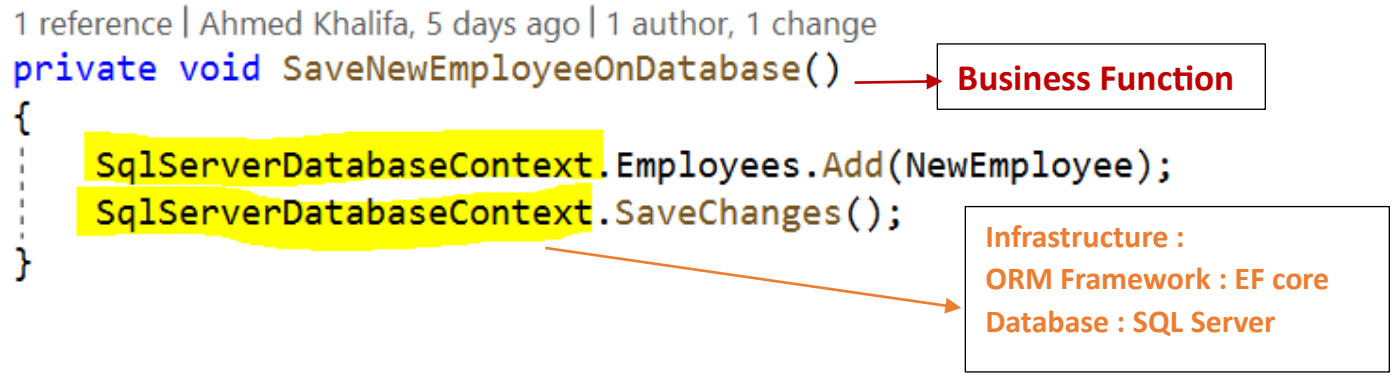


again in our business functions but this time we will discuss the infrastructure problem :

```
1 reference | Ahmed Khalifa, 22 hours ago | 1 author, 4 changes
private void CheckIfEmployeePhoneNumberIsRequiredAndUnique()
{
    if (string.IsNullOrEmpty(EmployeePhoneNumberTextBox.Text))
    {
        Errors.Add("Phone number is required. ");
    }

    bool isPhoneNumberExist = SqlServerDatabaseContext
                              .Employees
                              .Where(e => e.PhoneNumber == EmployeePhoneNumberTextBox.Text)
                              .Any();
    if (isPhoneNumberExist)
    {
        Errors.Add("Phone number is already exist , please enter unique phone number. ");
    }
}
```

Business Rule Function

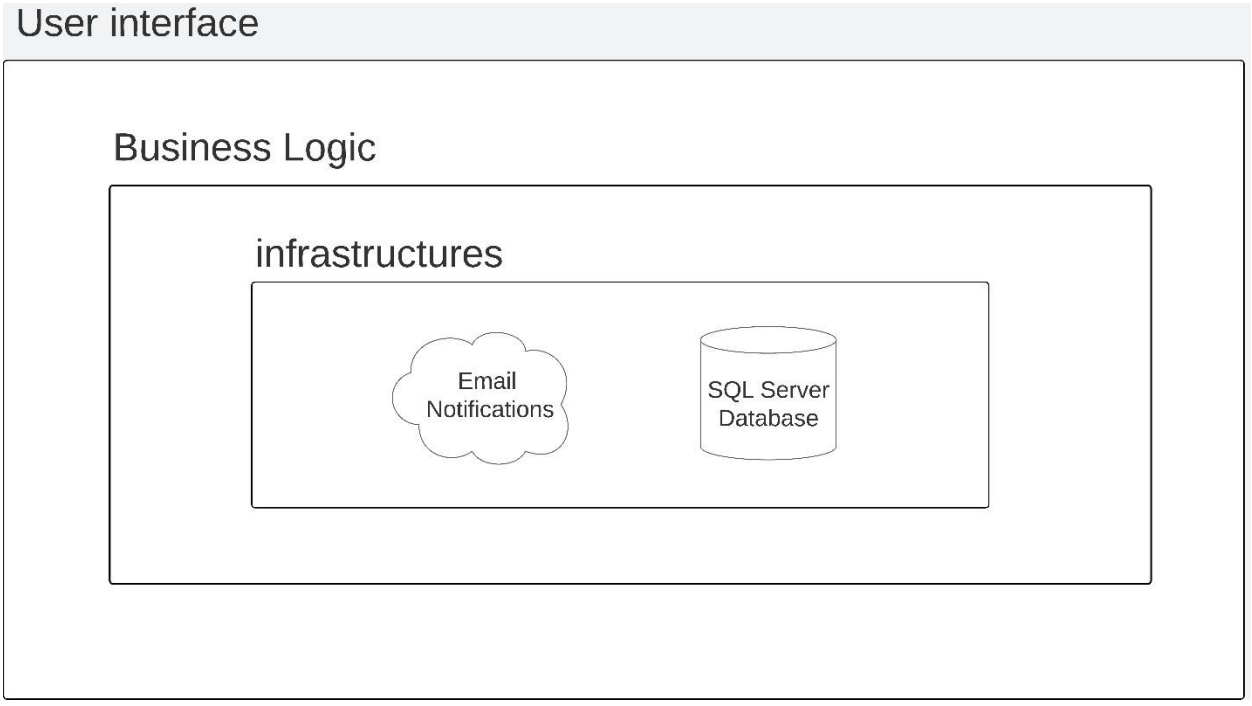
Infrastructure :
ORM Framework : EF core
Database : SQL Server



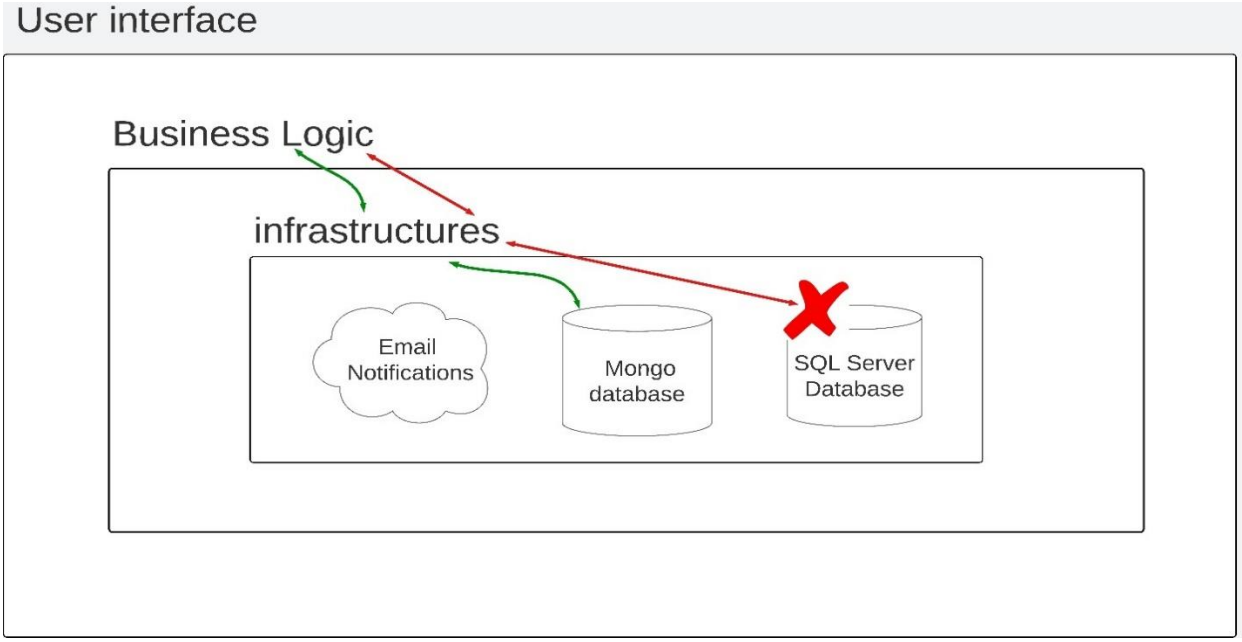
The two previous business functions contain **the actual implementation of the infrastructure code** for database , so the business logic will be very coupled on both the ORM framework and SQL Server database , so will have two cases :

1- changing the ORM framework from entity framework core to for example dapper or ado.net
2- changing the physical database from SQL Server to for example MongoDB or redis
this two cases or any one of them will have a big effect on business function or logic although there are no actual change on the business itself , **business function should only change when there are change on the business itself not on the infrastructure component**

This effect is happen because our code make the architecture look like this :



the outer layers or code will be very depended on the inner layers or code.
so any change on the ORM Framework or the physical database will automatically effect on the business logic.



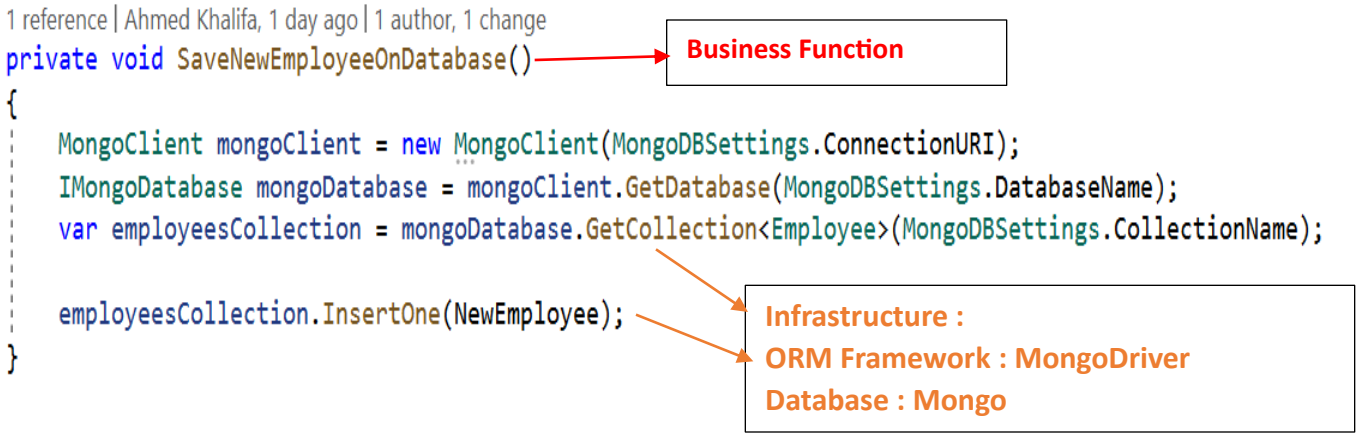
The new code after replacing SQL server by mongo database will modify the business functions and has a big effect on them :

1 reference | Ahmed Khalifa, 1 day ago | 1 author, 1 change

```
private void CheckIfEmployeePhoneNumberIsRequiredAndUnique()
{
    if (string.IsNullOrEmpty(EmployeePhoneNumberTextBox.Text))
    {
        Errors.Add("Phone number is required. ");
    }

    MongoClient mongoClient = new MongoClient(MongoDBSettings.ConnectionURI);
    IMongoDatabase mongoDatabase = mongoClient.GetDatabase(MongoDBSettings.DatabaseName);
    var employeesCollection = mongoDatabase.GetCollection<Employee>(MongoDBSettings.CollectionName);

    bool isPhoneNumberExist = employeesCollection
                              .CountDocuments(employee => employee.PhoneNumber == EmployeePhoneNumberTextBox.Text)
                              > 0;

    if (isPhoneNumberExist)
    {
        Errors.Add("Phone number is already exist , please enter unique phone number. ");
    }
}
```

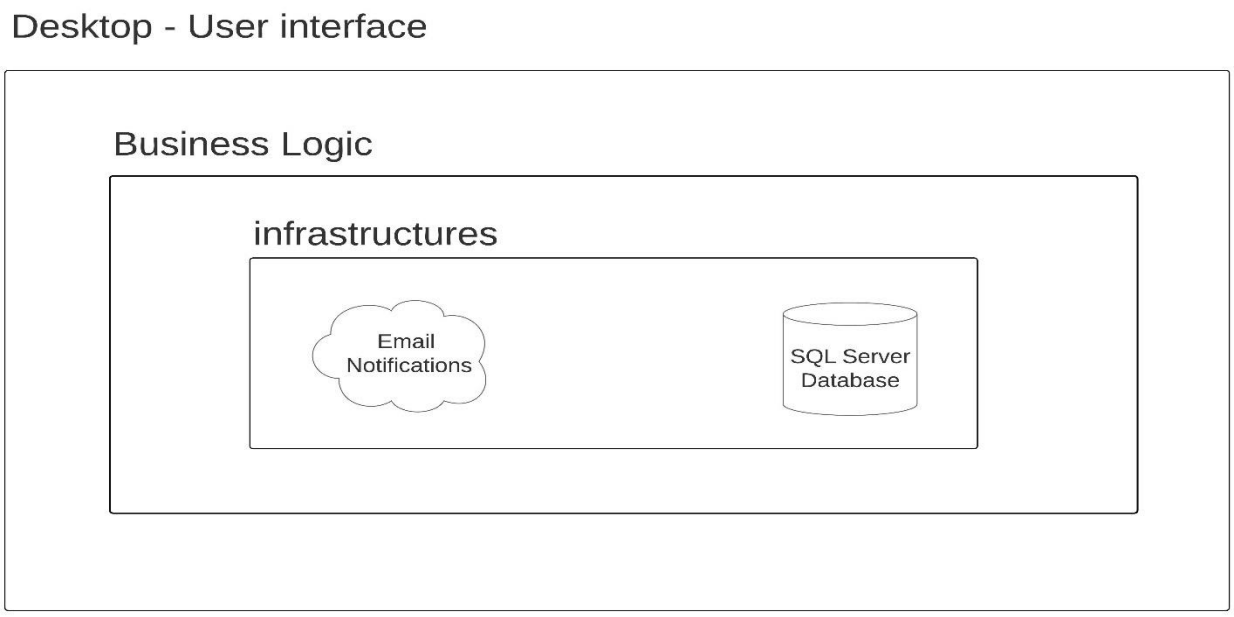
Business Rule Function

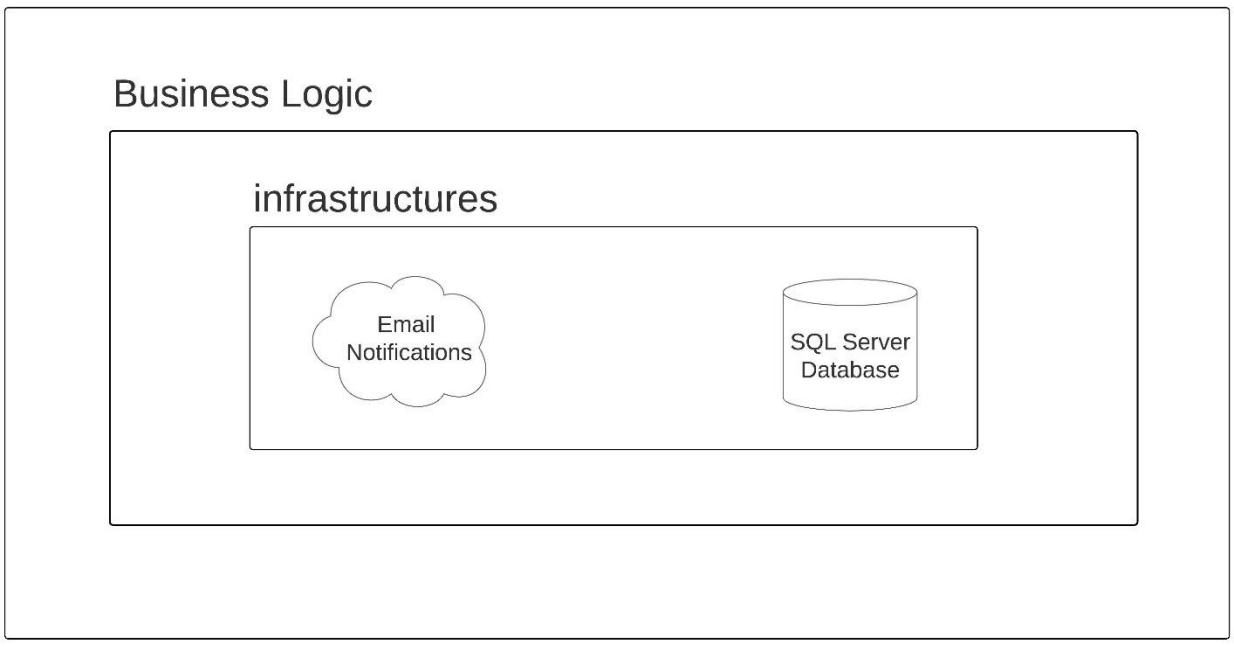
Infrastructure :
ORM Framework : MongoDriver
Database : Mongo


1 reference | Ahmed Khalifa, 1 day ago | 1 author, 1 change
private void SaveNewEmployeeOnDatabase()
{
MongoClient mongoClient = new MongoClient(MongoDBSettings.ConnectionURI);
IMongoDatabase mongoDatabase = mongoClient.GetDatabase(MongoDBSettings.DatabaseName);
var employeesCollection = mongoDatabase.GetCollection<Employee>(MongoDBSettings.CollectionName);
employeesCollection.InsertOne(NewEmployee);
}
Business Function
Infrastructure :
ORM Framework : MongoDriver
Database : Mongo

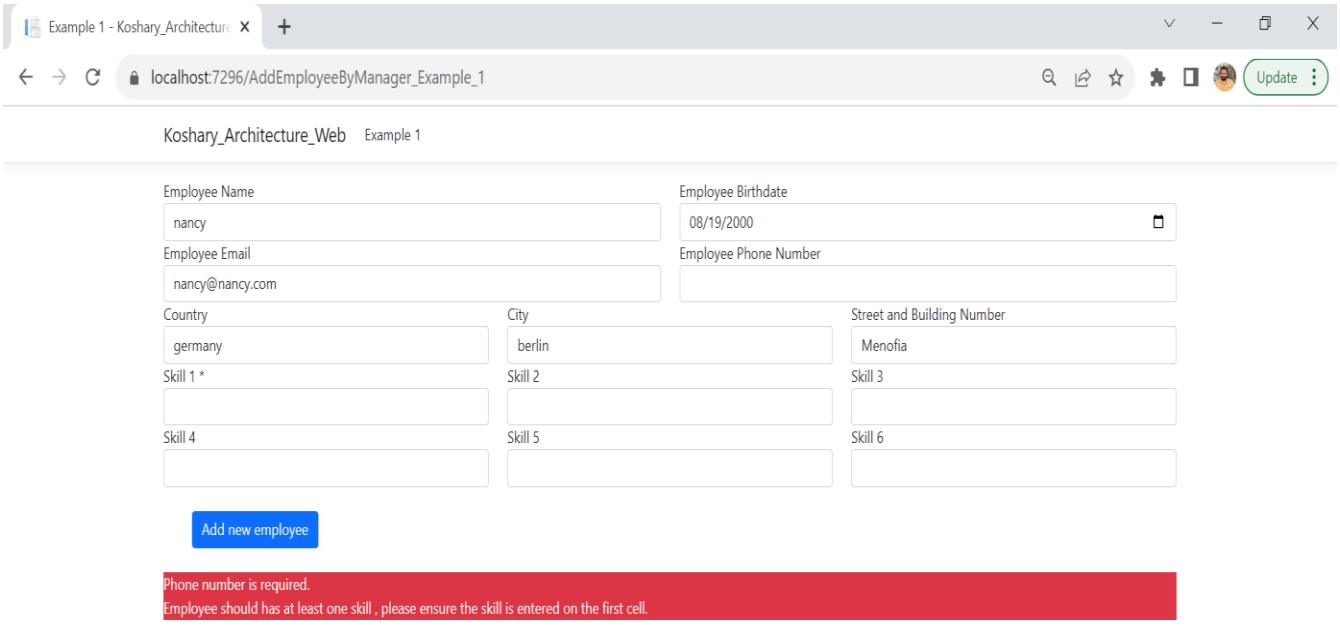
Fourth problem : Code Duplications

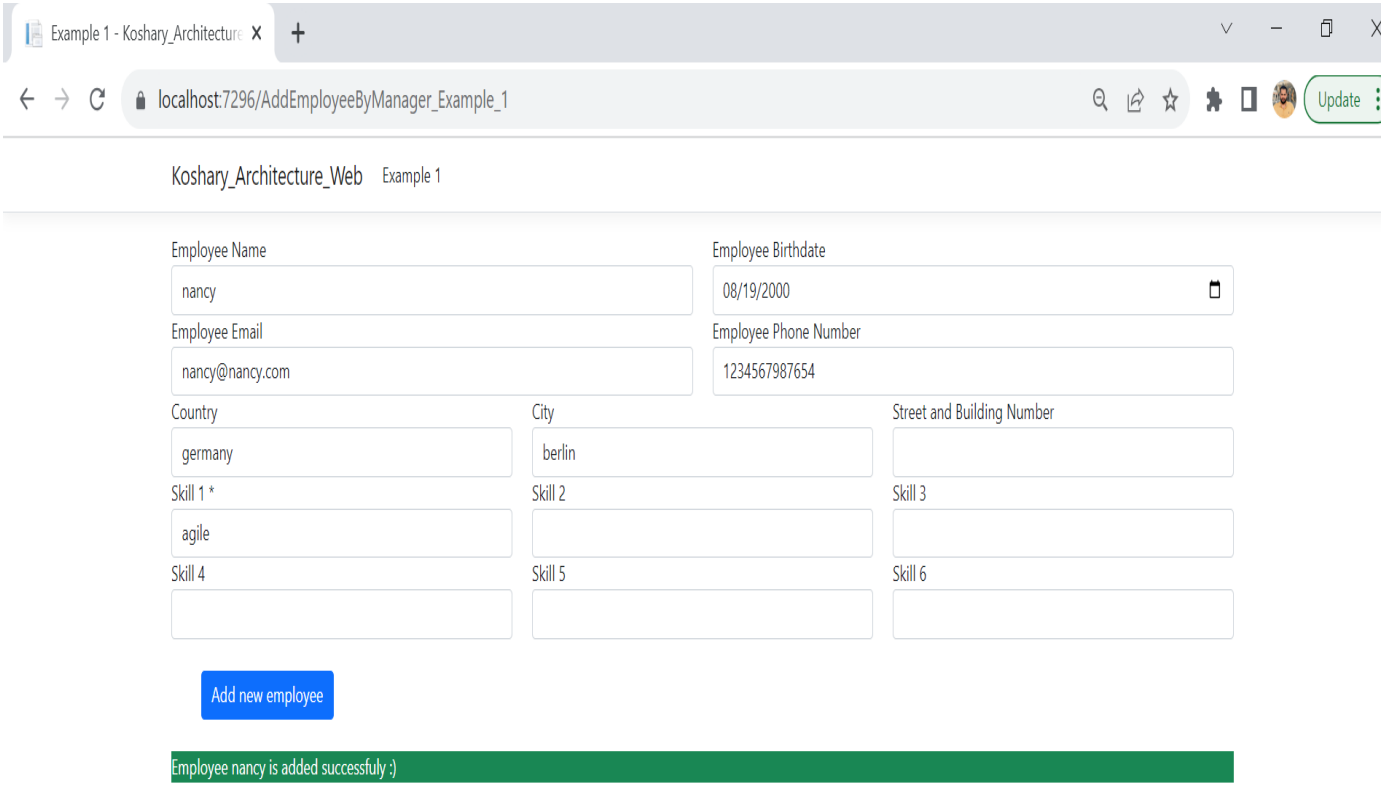
Now our company is growth and need web application beside the current desktop application, the two architectures will be like that :





The web application UI/UX :





Web application page model component code :

```
9   namespace Koshary_Architecture_Web.Pages
10  {
        5 references | Ahmed Khalifa, 1 day ago | 1 author, 3 changes
11      public class AddEmployeeByManager_Example_1_Model : PageModel
12      {
13          Properties
20
            0 references | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
21          public void OnGet()...
25
            0 references | Ahmed Khalifa, 1 day ago | 1 author, 1 change
26          public void OnPost()
27          {
28              InitUiConfiguration();
29
30              CheckIfEmployeeAgeIsEqualOrGreaterThan21Years();
31              CheckIfEmailHasCorrectFormat();
32              CheckIfEmployeePhoneNumberIsRequiredAndUnique();
33              CheckIfEmployeeHasAtLeastOneSkill();
34
35              if (Errors.Count > 0)
36              {
37                  DisplayErrorMessage();
38              }
39              else
40              {
41                  CreateNewEmployee();
42                  SaveNewEmployeeOnDatabase();
43                  SendEmailNotificationToAdmin();
44                  DisplaySuccessMessage();
45              }
46          }
47
            1 reference | Ahmed Khalifa, 1 day ago | 1 author, 1 change
48          private void InitUiConfiguration()
49          {
50              Errors = new List<string>();
51
52              if (NewEmployeeUserInterfaceModel == null)
53                  NewEmployeeUserInterfaceModel = new NewEmployeeUserInterfaceModel();
54          }
55
            1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
56          private void CheckIfEmployeeAgeIsEqualOrGreaterThan21Years()
57          {
58              DateTime today = DateTime.Now;
59              var age = today.Year - NewEmployeeUserInterfaceModel.Birthdate.Year;
60              if (age < 21)
61              {
62                  Errors.Add("Employee age should be equal or greater than 21 years.");
63              }
64          }
65
```

```
1 reference | Ahmed Khalifa, 1 day ago | 1 author, 3 changes
66  private void CheckIfEmailHasCorrectFormat()
67  {
68      bool isEmail = Regex.IsMatch(NewEmployeeUserInterfaceModel.Email,
69                                   @"\A(?:[a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)*
70                                   RegexOptions.IgnoreCase);
71
72      if (!isEmail)
73      {
74          Errors.Add("Employee email must has correct format.");
75      }
76  }
77
    1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
78  private void CheckIfEmployeePhoneNumberIsRequiredAndUnique()
79  {
80      if (string.IsNullOrEmpty(NewEmployeeUserInterfaceModel.PhoneNumber))
81      {
82          Errors.Add("Phone number is required. ");
83      }
84
85      bool isPhoneNumberExist =   SqlServerDatabaseContext
86                                .Employees
87                                .Where(e => e.PhoneNumber == NewEmployeeUserInterfaceModel.PhoneNumber)
88                                .Any();
89      if (isPhoneNumberExist)
90      {
91          Errors.Add("Phone number is already exist , please enter unique phone number. ");
92      }
93  }
94
```

```
1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
96  private void CheckIfEmployeeHasAtLeastOneSkill()
97  {
98      if (string.IsNullOrWhiteSpace(NewEmployeeUserInterfaceModel.EmployeeSkill_1))
99      {
100         Errors.Add("Employee should has at least one skill , " +
101                    "please ensure the skill is entered on the first cell. ");
102     }
103 }
104
```

```
1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
105     private void CreateNewEmployee()
106     {
107         NewEmployee = new Employee();
108         NewEmployee.Name = NewEmployeeUserInterfaceModel.Name;
109         NewEmployee.Birthdate = NewEmployeeUserInterfaceModel.Birthdate;
110         NewEmployee.Email = NewEmployeeUserInterfaceModel.Email;
111         NewEmployee.PhoneNumber = NewEmployeeUserInterfaceModel.PhoneNumber;
112         NewEmployee.Country = NewEmployeeUserInterfaceModel.Country;
113         NewEmployee.City = NewEmployeeUserInterfaceModel.City;
114         NewEmployee.StreetAndBuildingNumber = NewEmployeeUserInterfaceModel.StreetAndBuildingNumber;
115 
116         List<EmployeeSkill> employeeSkills = new List<EmployeeSkill>();
117         employeeSkills.Add(new EmployeeSkill { SkillName = NewEmployeeUserInterfaceModel.EmployeeSkill_1 });
118         if (!string.IsNullOrEmpty(NewEmployeeUserInterfaceModel.EmployeeSkill_2))
119             employeeSkills.Add(new EmployeeSkill { SkillName = NewEmployeeUserInterfaceModel.EmployeeSkill_2 });
120         if (!string.IsNullOrEmpty(NewEmployeeUserInterfaceModel.EmployeeSkill_3))
121             employeeSkills.Add(new EmployeeSkill { SkillName = NewEmployeeUserInterfaceModel.EmployeeSkill_3 });
122         if (!string.IsNullOrEmpty(NewEmployeeUserInterfaceModel.EmployeeSkill_4))
123             employeeSkills.Add(new EmployeeSkill { SkillName = NewEmployeeUserInterfaceModel.EmployeeSkill_4 });
124         if (!string.IsNullOrEmpty(NewEmployeeUserInterfaceModel.EmployeeSkill_5))
125             employeeSkills.Add(new EmployeeSkill { SkillName = NewEmployeeUserInterfaceModel.EmployeeSkill_5 });
126         if (!string.IsNullOrEmpty(NewEmployeeUserInterfaceModel.EmployeeSkill_6))
127             employeeSkills.Add(new EmployeeSkill { SkillName = NewEmployeeUserInterfaceModel.EmployeeSkill_6 });
128 
129         NewEmployee.EmployeeSkills = employeeSkills;
130     }
131 
```

```
        1 reference | Ahmed Khalifa, 3 days ago | 1 author, 1 change
132     private void SaveNewEmployeeOnDatabase()
133     {
134         SqlServerDatabaseContext.Employees.Add(NewEmployee);
135         SqlServerDatabaseContext.SaveChanges();
136     }
137
        1 reference | Ahmed Khalifa, 3 days ago | 1 author, 1 change
138     private void SendEmailNotificationToAdmin()
139     {
140         Console.WriteLine("Email is sent ...");
141         Console.WriteLine("you can uncomment the code below to send actual email ");
142         /*
143         https://www.c-sharpcorner.com/article/sending-email-using-c-sharp/
144         MailMessage message = new MailMessage();
145         SmtpClient smtp = new SmtpClient();
146         message.From = new MailAddress("FromMailAddress");
147         message.To.Add(new MailAddress("ToMailAddress"));
148         message.Subject = "Test";
149         message.IsBodyHtml = true; //to make message body as html
150         message.Body = htmlString;
151         smtp.Port = 587;
152         smtp.Host = "smtp.gmail.com"; //for gmail host
153         smtp.EnableSsl = true;
154         smtp.UseDefaultCredentials = false;
155         smtp.Credentials = new NetworkCredential("FromMailAddress", "password");
156         smtp.DeliveryMethod = SmtpDeliveryMethod.Network;
157         smtp.Send(message);
158         */
159     }
160
        1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
162     private void DisplaySuccessMessage()
163     {
164         NewEmployeeUserInterfaceModel.IsSuccess = true;
165
166         string successMessage = "Employee " + NewEmployee.Name + " is added successfuly :)";
167         NewEmployeeUserInterfaceModel.Message = successMessage;
168     }
169
        1 reference | Ahmed Khalifa, 1 day ago | 1 author, 2 changes
170     private void DisplayErrorMessage()
171     {
172         NewEmployeeUserInterfaceModel.IsSuccess = false;
173
174         string errorMessage = string.Empty;
175         foreach (var error in Errors)
176         {
177             errorMessage += error + "<br/>";
178         }
179         NewEmployeeUserInterfaceModel.Message = errorMessage;
180     }
181 }
182 }
```

The code in two project will be duplicated , duplication is the evil of the software.

for example if you want to change business rule from min employee age 21 to 18 , the developer need to duplicated the code again in the two projects.

Desktop :

```
1 reference | 0 changes | 0 authors, 0 changes
private void CheckIfEmployeeAgeIsEqualOrGreaterThan18Years()
{
    DateTime today = DateTime.Now;
    var age = today.Year - EmployeeBirthDatePickerBox.Value.Year;
    if (age < 18)
    {
        Errors.Add("Employee age should be equal or greater than 21 years.");
    }
}
```

Web :

```
1 reference | 0 changes | 0 authors, 0 changes
private void CheckIfEmployeeAgeIsEqualOrGreaterThan18Years()
{
    DateTime today = DateTime.Now;
    var age = today.Year - NewEmployeeUserInterfaceModel.Birthdate.Year;
    if (age < 18)
    {
        Errors.Add("Employee age should be equal or greater than 21 years.");
    }
}
```