



**POLITECNICO
DI TORINO**

Computer Engineering
POLITECNICO DI TORINO
Italy

Development of a new user interface for an access control system

Abel Berhanu Zebre

Advisors

Prof. Prinetto Paolo Ernesto

Sig. Giuseppe Migliasso (Company)

Abstract

Developing a new user interface for any web application is a task that requires a deep understanding on the requirements such as, the project's concept, brainstorming, sketching, designing user flow diagrams, prototyping, evaluation, and validation. There might be also other necessary stages and tasks that should be addressed depending on the type and size of the project.

Microntel Access control system is a web application to manage employees, visitors, badges and keys, access stamps, canteen, reports, and authorization issues that can be used by any client company. The system was developed before 15 years and now the company decided to refactor it with a new user interface by using the latest technology tools so that the system would function in the efficient way possible.

This work lies in the realm of a bigger project aimed at studying the overall structure of both the old and the new project to identify the changes that are going to be done and involve in some part of the application. My work focused on clarifying the difference between the two projects by comparing the old version with the new version stressing the significant effect in the improvement process.

In my work I used the latest development and state-of-the-art technologies for the UI to improve the performance, flexibility, efficiency and visual appearance of the application.

Acknowledgements

I would first like to thank Prof. PRINETTO PAOLO ERNESTO and Dr. AIRO' FARULLA GIUSEPPE for the patience, motivation, and for the continuous guidance throughout my thesis work.

I would also like to thank the owners, the developers, the managers, the whole team in Microntel company. Without their participation and input, my work could not have been successfully conducted.

Finally, I must express my very profound gratitude to God and my families for providing me with unfailing support and continues encouragement throughout my years of study and through the process of researching and writing this thesis. Thank you.

Contents

1	General Introduction	8
1.1	Introduction	8
1.2	Access control system	9
1.2.1	Access control panel	9
1.2.2	Access control reader	10
1.2.3	The device being detected	11
1.2.4	The software	11
1.2.5	The entry and exit hardware	12
1.3	Thesis structure	13
2	The Old system application	14
2.1	Introduction	14
2.2	Sub-Systems	16
2.2.1	MicronSin	16
2.2.2	MicronPass WEB RealTime	16
2.2.3	MicronWebService	16
2.2.4	MRT DataBase	18
2.3	Work flow	19
2.3.1	Guest	21
2.3.2	Visitor booking	23
2.3.3	List of Visitor bookings	24
3	The New system application	25
3.1	Architectural overview	25
3.2	ASP.NET MVC entity framework	30
3.2.1	Entity framework	31
4	New technologies	33
4.1	Introduction	33
4.2	Practical explanation	36
4.2.1	Code comparison	36
4.3	Knockout.js	44
4.3.1	JSON	45
4.3.2	Sample effects of the new technologies	46
4.3.3	Filtering description	48

5 Comparison of the two systems	51
5.1 Time slot	51
5.1.1 Name Space Comparison	52
5.1.2 Code comparison	55
6 Unit testing	59
6.1 Introduction	59
6.1.1 Advantages of unit testing	60
6.2 Sample unit test	62
6.2.1 Unit test result	64
7 Conclusion and Future Work	67
7.1 Future work	68
7.2 Conclusion	68

List of Figures

1.1	Access Control Panel	9
1.2	Access Control Reader	10
1.3	Device being detected	11
1.4	Entry Hardware	12
2.1	IIS configuration in widows operating system	15
2.2	Microntelplants	17
2.3	Custom loading image	19
2.4	Log-in page	19
2.5	Home page	20
2.6	Guest page	21
2.7	Adding guest page	22
2.8	Visitor booking	23
2.9	View Booking	24
3.1	Old system architectural view	26
3.2	New system architectural view	27
3.3	New system architectural view with multiple frameworks	28
3.4	New system architectural with multiple MVC application	29
3.5	Model-View-Controller	30
3.6	View	31
3.7	Repository	32
4.1	Illustration of Ajax suggestion method used by google	36
4.2	Simple addition of two numbers index code	37
4.3	Simple addition of two numbers Ajax index code	38
4.4	Controller page of the normal ASP.	39
4.5	Controller page of the Ajax Controller.	40
4.6	The Interface of addition of two numbers	41
4.7	Inspection Ajax response body	42
4.8	Inspection Ajax response body	43
4.9	Time slot edit page	46
4.10	Binding prompt	47
4.11	Filtering by code and description	48
4.12	Filtering by Plant	49
4.13	Filtering by logical seat	49
4.14	Filtering by Technology	50
4.15	Filtering by Gate or group	50

LIST OF FIGURES

5.1	Sample Comparison when accessing the database	55
5.2	Comparison Summary of Old (left side) and new (right side) systems	57
6.1	New booking form	63
6.2	Unit test for First-Name	64

Chapter 1

General Introduction

1.1 Introduction

A user interface (UI) is the crucial part of a system because it is important to the general user experience of a system application since a user is obligated to use the user interface to interact to the system.

Developing this user interface which is going to be the bridge between the system and the user is the very crucial part of a system in regarding the commercial, functionality, sustainability of the whole application. The development of a new user interface for an already working application is the same as developing a new user interface from scratch but without the ability to touch the skeleton structure of the system.

Re-factoring a web application is very restrictive on the freedom of applying extra futures that might touch or affect the database or other basic structure of the whole system. Because of this reason its main intention is to improve the code quality without changing the main structural functionality.

There are usually two practices when renewing a user interface. Refactoring the user experience and User interface Refactoring. One is more concerned with the experience of the user and the other concerns more on the improvement of the UI itself.

To adequately adopt user satisfaction on the process of refactoring the user interface, one should consider understanding the users which includes identifying what types of users are using the system (age, education level. . .), it also need to consider what the user want to achieve with an existing system but also not forgetting what they might want in the future.

When improving the UI, the main aim is to make it simple as much as possible and It should be easy or intuitive. But it also needs to include all basic and valuable features in a straightforward manner.

1.2 Access control system

Access control is the combination of hardware and software that controls access to an entry point of a secure structure or property. It can be used for commercial or residential use. The purpose of this product is to manage and grant access to individuals to specific areas on a predetermined schedule. An example of access control can be an automated gate in front of a residential community or a company building.

There are different varieties of access control types when accessing a restricted area. The access can be granted by entering a correct password on an access control device, or a card can be used which has a chip that holds the user information, or a biometric method can be used which scan and read the user's eye, hand or fingerprint.

1.2.1 Access control panel

An access control panel is the brain of all components. They are made of a circuit inside a closed box along with power supply and sometimes with a backup battery supply. Fig 1.1 The number of access points or doors depends on the capability of this circuit board.



Figure 1.1: Access Control Panel

1.2.2 Access control reader

These readers can detect a person who has a compatible device on them such as an access card or key. Depending on the equipment it can detect device at variant distances. In some cases the user needs to swipe the card in other cases it might detect from distances. Unlike proximity readers keypad requires type and access code directly into the unit. The other type of reader can be categorized as biometric readers. Fingerprint reader, facial detection reader, iris detection reader or other. This can recognize the user figure print, face, or eye to determine appropriate access for that person. There are also long-range readers that can detect the user from a long distance. see figure 1.2

From the many types of access control readers, the reader that also manages the entry time is the vital one which is also one of the services that Microntel Company offers. This reader records when the employee is in and out. Ad advanced feature is available with the addition of time management software.



Figure 1.2: Access Control Reader

1.2.3 The device being detected

There are multiple types of access control cards like smart cards, long distance cards or key fobs. Most commonly used detection devices are smart/long distance cards. They can come in many forms. All of them holds information about the user and their access rights. They communicate with the reader through radio frequency. The range that the cards can communicate and the amount of information they can hold depends on the cards specification. A smart card is also used to hold the same information but with greater capacity. Smart card holds information like medical information or even they can be used as a debit card. Fobs are the same as access cards or it can be used as a keychain. Some of them have even sounds that alert the users if they granted or denied access. An alternative to access cards, key fobs or stickers can be put on the back of a phone or any other objects that some can carry around most of the time.



(a) smart card



(b) key fobs

Figure 1.3: Device being detected

1.2.4 The software

Finally, there is the software. Although many access control system comes with integrated software, some may require additional software to run more additional features. The software is where we manage users, users rights, schedules and more. Many systems have add-on software like time and attendance, visitor management and video management. Here is where we are focusing on.

1.2.5 The entry and exit hardware

Once the user granted access it must send a signal to unlock it. In this case, the door must have special equipment to unlock it that works with the access control system. The two most mechanisms are electric door strikes and mag locks. electric door strikes replace the ordinary door strike. They are electric powered and automatically unlock when the user is granted access.

Magnetic locks are electric powered magnets. This lock will hold the door locked until the system sends the signal to unlock the door.

Exit buttons and bars usually go in the other side of the door and allow the user to exit from a secure area. There are many types of exit button and bars. Some are simple electric buttons and some are nomadic. The nomadic one does not need any power. So an exit is possible even when power is not available. There are also exit buttons with a delay from more secure areas when the user must wait for few seconds before exiting is possible.



Figure 1.4: Entry Hardware

1.3 Thesis structure

This thesis is structured in a way that it can be used as a reference for any re-factoring process of a user interface in Microsoft environment. It describes different areas and modules, improvements, technologies used in the process of the development of a new user interface.

Chapter (2) describes the architectural design options chosen for the old system. The chapter describes what are the necessary tools to develop the system and tries to highlight how they are configured, used and list out some of their importance. The chapter also describes some of the subsystems that are used in the overall system with a graphical representation of their interaction with each other including how they use the same database. The chapter finally shows the system work flow describing each scenario and the interaction that takes place when the application is used with illustrations taking the "guest page" as an example.

Chapter (3) is all about the new system structure to give a good overview of the new system with the description on how the new system will improve the old system by giving a general structural view. The chapter also describes the programming language used in the development process and how it upgrades the programming language that had been used in developing the old one.

Chapter (4) is about the new technologies that have been used in the new system. Since most of the client side scripts are discovered in recent years, these technologies were not available at the first implementation of the system. The chapter describes these technologies, what they mean and their advantages in the improvement process of the system. In addition, the chapter describes how these newly added technologies influence positively the system by showing their effect on a single sample functionality called "Time slot".

Chapter (5) describes the difference between the old and the new application development by narrating what exactly is improved inside the code. By taking the "time slot" page as an instance, it shows what is happening behind the scenes and what it means to renew an application concerning the code. It clearly shows the changes on the user interface as well as the changes in the process of development of the system.

Chapter (6) is about unit testing. It shows what exactly means to use unit testing, their methodologies, why it is needed specially in a big application and finally providing a real time scenario as an example for the system.

Finally Chapter (7) discusses what can be done in the future, and the conclusion .

Chapter 2

The Old system application

2.1 Introduction

Every web application system that has been developed using Microsoft.Net framework needs some configuration to be done. Before deploying a system into a user computing machine there are a few things that need to be configured in order to make the system work. There are many choices to host a system. We can host it using FTP,¹ or we can use some data-center or we may also host on Microsoft Azure (just assume it as a large data center running in off site)² But since our system is configured on IIS (Internet Information System), we will focus on how this configuration works.

To configure the IIS we should set up the IIS management console and ASP.NET components from “Turn Windows features on or off” tab of the control panel in a windows operating system. There are many advantages of using IIS to host a system. Some of them are.

- Reduced cost of deployment for users: To run an application built with IIS, the user of the application can run the application using only a browser; no special third party software is needed to be installed on their computer to run the application.
- Access to a broad audience: IIS applications works with a wide variety of browsers and operating systems, thus we can easily reach a wide audience.
- Separation of code and HTML: Unlike scripting (written for a run-time environment), our code is not embedded in the HTML document, so we can separate the process of designing user interface from writing, testing, and debugging code.
- The other advantage could be State management across multiple interactions with the client: we can manage state using objects or a database, or even we can shuttle state between the client and the server.

¹ a file transfer protocol that is used to transfer data between computing machines

² a collection of resources owned by Microsoft that can be used to provide management of applications on the internet.

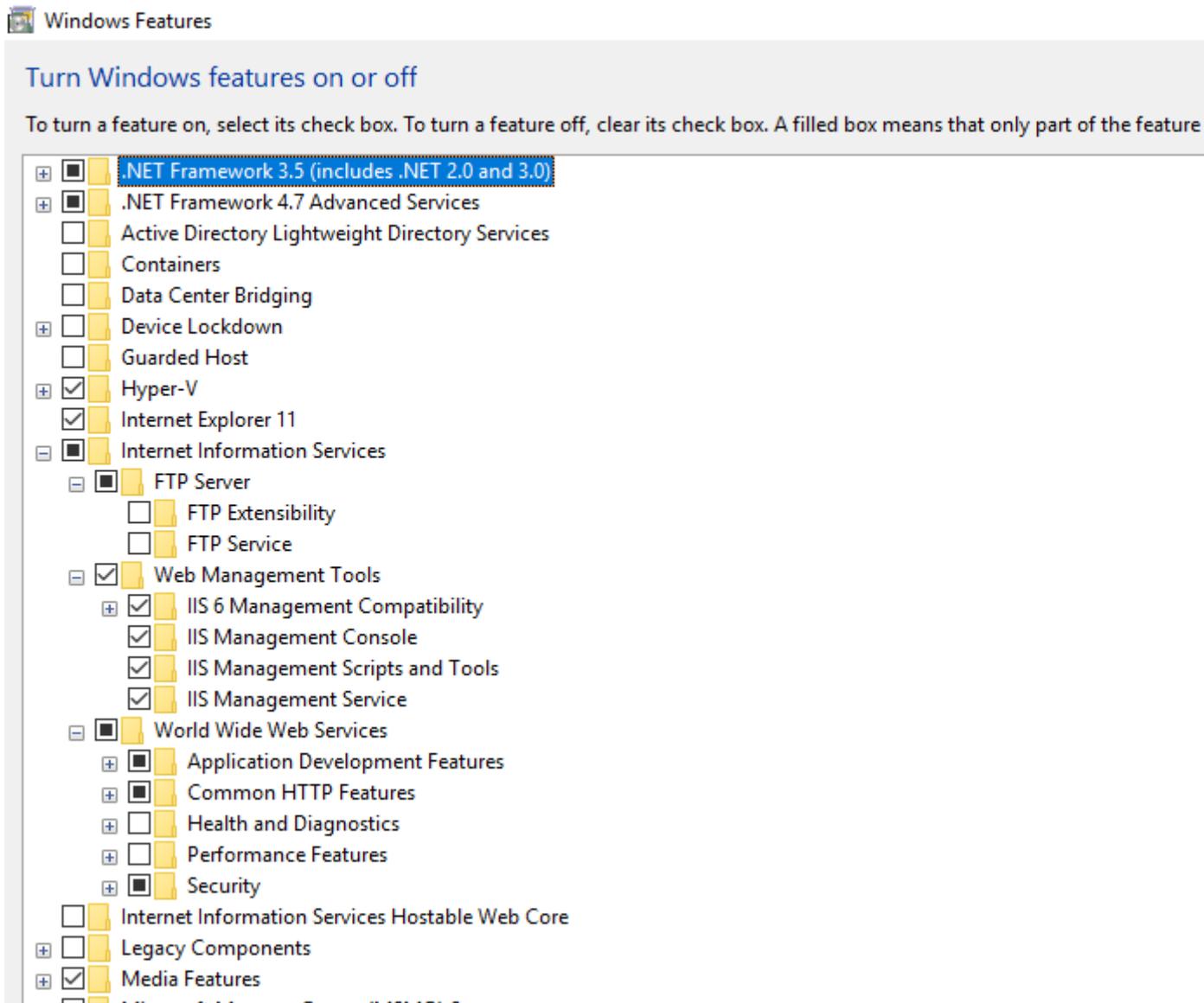


Figure 2.1: IIS configuration in windows operating system

Once the IIS is configured, the next step would be to set up the database. A database is in its simple form: a space of memory that used to store data which then enables a user to retrieve and manipulate the data in an effective and efficient way. A database management system (DBMS) is used for the management of this data.

DBMS[see site 1] is a collection of programs that manages database structure and controls access to the data stored in the database. In a sense, a database resembles a very well-organized electronic filing cabinet in which powerful software, known as a database management system, helps manage the cabinet's contents. The DBMS serves as the intermediary between the user and the database.

The database structure itself is stored as a collection of files, and the only way to access the data in those files is through the DBMS. The DBMS receives all application requests and translates them into the complex operations required to fulfill those requests. The DBMS hides much of the database's internal complexity from the application programs and users. There are so many advantages of DBMS. some of them are; improving data sharing, improving data security, better data integration, minimizing data inconsistency, improving data access, improving decision making, and so on.

finally after configuration of the database and the IIS, we should deploy all needed system data and all necessary components to an empty folder so that we can add it to the IIS setup. Once we setup all this, we can browse our web system using the URL specified in the IIS. Or we can browse the system from the IIS page.

2.2 Sub-Systems

In the over all company system there are subsystems that interact with each other that might be found online real-time or offline version. Even though these systems use the same database and interact with each other, each of them has different purpose and functionality. from the following listed subsystems, we are discussing in this paper the system "MicronPass".

2.2.1 MicronSin

Monitors the system by using colored icons to map and indicate the interactive gates and allows the visualization of the information like availability and status of gates and Karpos terminals, presence in the antipassback areas, and alarm with reset functions.

2.2.2 MicronPass WEB RealTime

This one is the main functionality of the system. It manages access profiles designation, to control gates and employees, with the authorization to access selected reports in relation to the organization chart. It also manages Employees ID which used to control all issues related to Employee ID.

2.2.3 MicronWebService

Is a Web service for the integration of personal data with external procedures. It allows the external management of the personal data (employees, external collaborators, guests) and the assignment of a peripheral access profile. With MicronWebService, it is possible to manage the importation of data in real time and manage the import of data in Real-time. It also has interactions with System interface like Human Resource, Building management, and others.

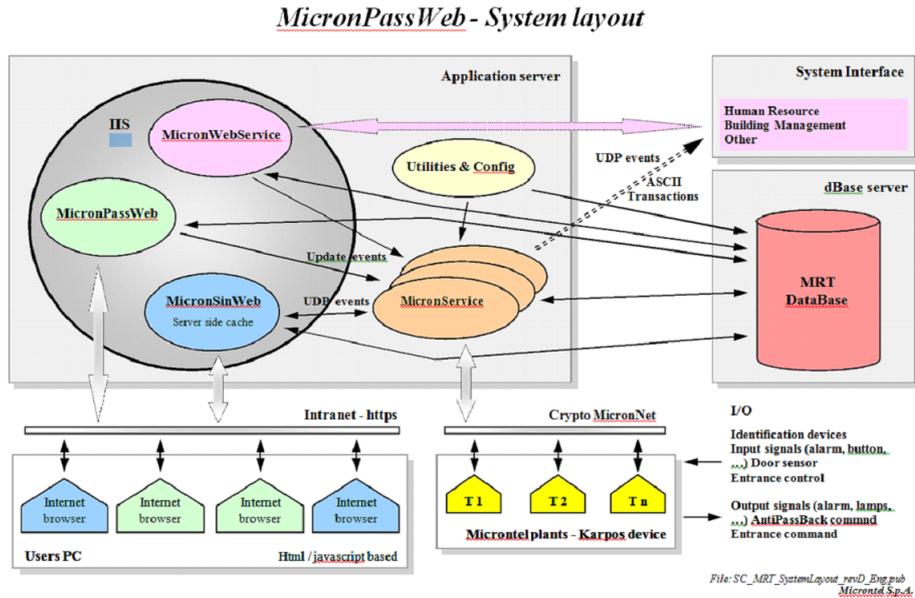


Figure 2.2: Microntelplants

This plant is composed of multiple identification devices which might be input signals or output signals. The input signals might be an alarm, button or other equipment that are used as a door sensor. On the other hand, the output signals might be alarms, lamps or other items that are used as antipassBack command.³

The Utilities and configuration help to analyze, configure, optimize or maintain the MicronService. After the management of the service is done, the MicronService can access the MRT database.

The user PC accesses the IIS (internet information service) system by using any internet browser. The browser will access the system using internet https which makes all the communication between the browser and the system site will be encrypted.

³ a method to control a card holder from passing his/her card to a third person

2.2.4 MRT DataBase

The MRT database is a collection of database that are used to fetch data from multiple subsystems from all over Microntel company systems and organize them so that it the data can be easily accessed, managed, and updated. [[2]]

The MRT database will be accessed by the application server through MicronSinWeb, MicronPassWeb , MicronWebService, Utilities and configuration and MicronService. But in the coming new system, the database will be accessed only by one gateway.

In this old architecture all the components on the left side are accessing the MRT DataBase which is not recommended because, in such type of design, there will not be a centralized component which manages the accessing of the MRT Database.

The new structure of the system will solve this problem by introducing a new feature in the middle which organizes the flows between the Database and the other system.

Though the functionality and architecture of the old system work fine, there are many things that can be improved from the graphic design, the technology used - to the functionality. As a reason, a new system is needed to improve all the defects of the old system.

The biggest problem with the old system in the process of re-factoring is the fact that we can not change the database and the basic skeleton of the system.

The other big problem is, naming problem. For instance the table names in the database was assigned by code not by name as a reason it was hard to know instantly to identify the tables. There is always a need of some consultation from the developers of the old application.

2.3 Work flow

The procedure and the graphical view of most of the system will not be changed in the new system. but there will be a significant change in the code to improve the old one regarding efficiency, speed, effectiveness, and functionality.

The first thing we see when we launch the application is the custom chrome loading motion image. This is a custom made to have a better graphical image to have a more interesting interface while it loads for the first log-in page of the system

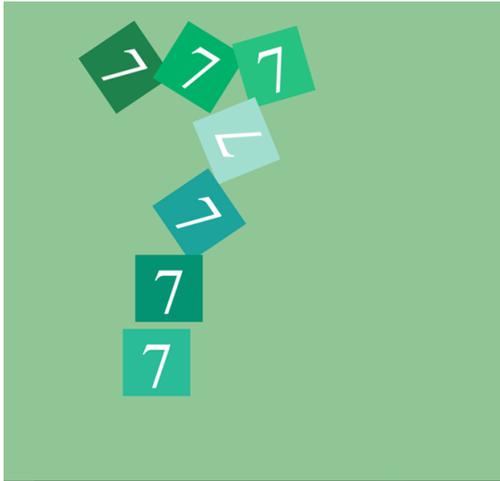


Figure 2.3: Custom loading image

After the user inserts the correct user and password, the system will allow accessing the main page. From this main page, the user can navigate to any section of the system or log-out.



Figure 2.4: Log-in page

The main page includes most of the navigation options. From this page, a user can navigate through the tabs/links at the left side or we can also do the same task by using right tabs with image representation.



Figure 2.5: Home page

For the sake of simplicity, we will see only the functionality of the Guest unit from the listed sections.

When a user mouse over around the “Guest” tab, the system will provide with possible suggestions. Guest, Guest template, self-check guests, Guest reception, Visit types, Visit booking, view booking, and courier’s management are the possible suggestion.

2.3.1 Guest

In this section, we can get the guests who visited the company. There are possibilities to filter using the guest last name, code and also we can list out guest with assigned badge or guests modified by an old user by checking the checklist. In this page, we also find links to add a guest or view booking

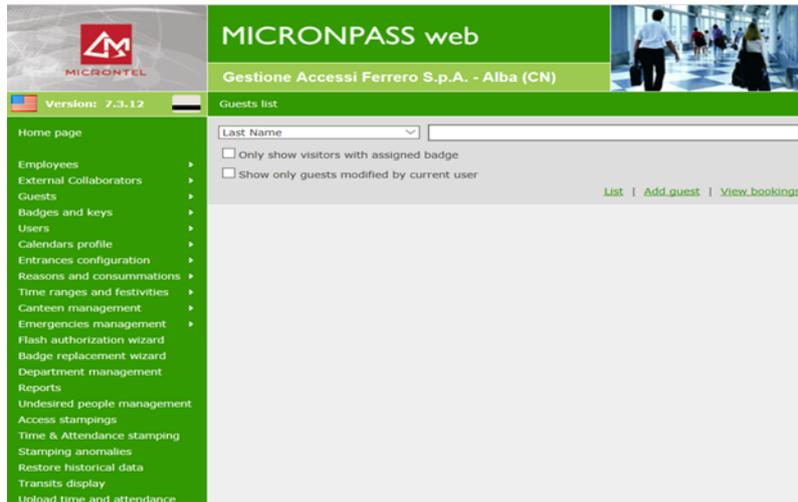


Figure 2.6: Guest page

The screenshot shows a web application interface for 'Gestione Accessi Ferrero S.p.A. - Alba (CN)'. The left sidebar contains a navigation menu with various options like 'Home page', 'Employees', 'External Collaborators', 'Guests', 'Badges and keys', 'Users', 'Calendars profile', 'Entrances configuration', 'Reasons and consummations', 'Time ranges and festivities', 'Canteen management', 'Emergencies management', 'Flash authorization wizard', 'Badge replacement wizard', 'Department management', 'Reports', 'Undesired people management', 'Access stampings', 'Time & Attendance stamping', 'Stamping anomalies', 'Restore historical data', 'Transits display', 'Upload time and attendance data', 'Attendance in real time', 'Current connections', and 'Change password'. The main content area is titled 'New guest' and contains the following form fields:

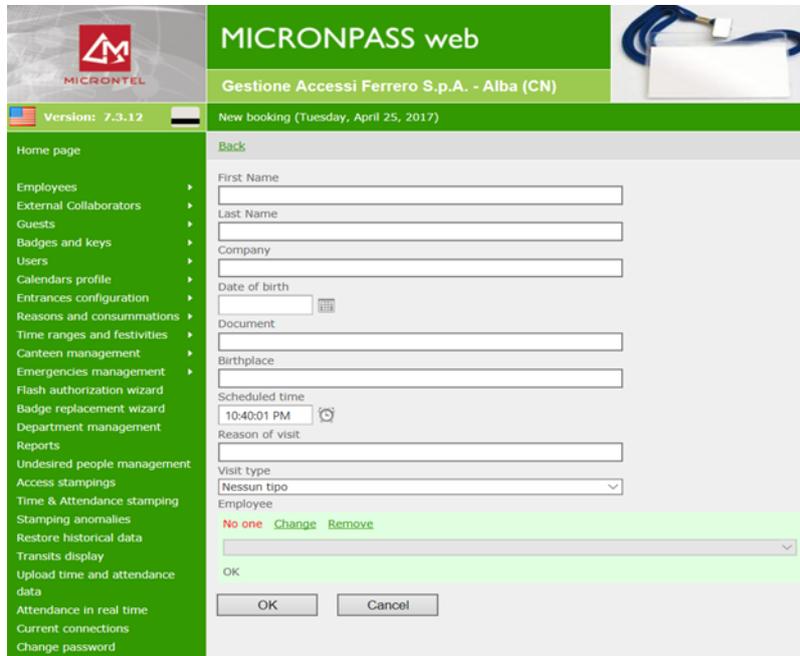
- Code (0=Automatically generated): 0
- First Name: [Empty]
- Last Name: [Empty]
- Company: [Empty]
- Active:
- Start date and time: Saturday, January 1, 2000 12:00:00 AM
- End date and time: Friday, December 31, 9999 11:59:59 PM
- Reference employee: 001 (with 'List' and 'Cancel' buttons)
- OK
- Visit type: No one
- Remarks: [Empty]
- Document: [Empty]
- Date of birth: 1/1/2000

Figure 2.7: Adding guest page

When adding a guest we can also use a predefined guest template so that we don't fill the whole guest form every time we are adding a guest.

2.3.2 Visitor booking

A visitor can book for a visit filling a simple form specifying a first name, Last name, company, date of birth, document, birthplace, schedule time, a reason of visit, visit type, and the employee to be visited



The screenshot displays the MICRONPASS web application interface. The header includes the MICRONTEL logo, the text "MICRONPASS web", and "Gestione Accessi Ferrero S.p.A. - Alba (CN)". A navigation menu on the left lists various system functions. The main content area is titled "New booking (Tuesday, April 25, 2017)" and contains a form with the following fields:

- First Name
- Last Name
- Company
- Date of birth
- Document
- Birthplace
- Scheduled time (set to 10:40:01 PM)
- Reason of visit
- Visit type (dropdown menu, currently set to "Nessun tipo")
- Employee (dropdown menu, currently set to "No one")

At the bottom of the form, there are "OK" and "Cancel" buttons.

Figure 2.8: Visitor booking

2.3.3 List of Visitor bookings

By stating start date, end date, and last name we can view the list of bookings. The user can also have an option to select not confirmed bookings or old user bookings.

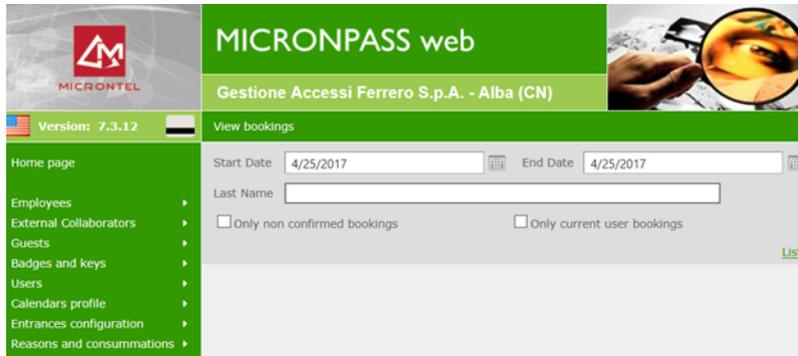


Figure 2.9: View Booking

Chapter 3

The New system application

3.1 Architectural overview

The new development of the system uses most of the old system data. Even some functionality will be reserved and unchanged. This is because developing a brand new web system application will require huge time, effort and cost. In addition, the database schema works fine as it is. Because of this reason the new development of the user interface will be based on the old system.

The new development of user interface for the access control system introduces many new technology tools including the most recent technologies like Javascript, JQuery, knockout and also other new system architectures.

Introducing such architecture will make it more effective, flexible, and easy to use. It also consumes less bandwidth of the network and it will ease the development process.

In the old system, each component of the services such as MicronPassWeb, MicronConfig, or other services have all the architectural layers (i.e application layer, business layer, and data layer).

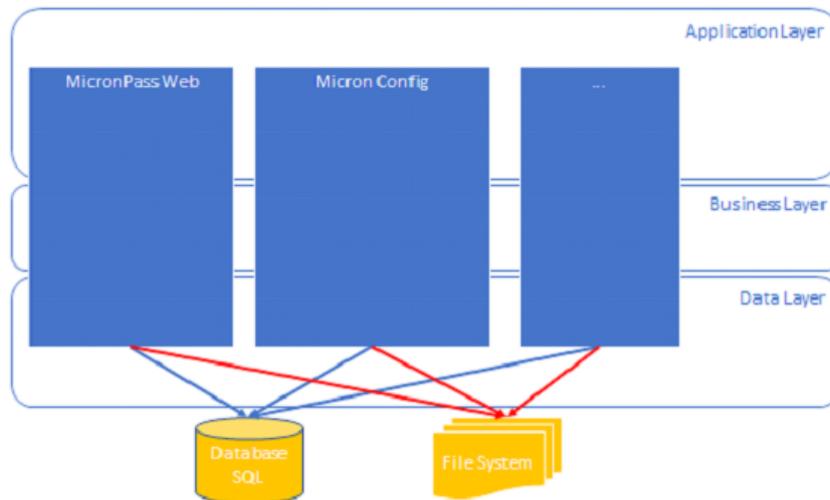


Figure 3.1: Old system architectural view

Such type of architecture is not encouraged because every time the developer wants to change something, they need to be a change in all of the layers which are so tiresome. The other defect of such structure is that it is very difficult to add additional features on the top of the system. With this architecture, there is no flow of interaction between the layers. And also in the development of this type of phenomena the process is slow, it is hard to implement and more coding is required to accomplish a complex task. It is also difficult to map business objects and this affects highly on the performance of the system.

- **Application layer:** This layer deals with the work on the user interface. Since ASP.Net MVC provides a modular architectural pattern, the development of user interface can be done in this application layer without any dependency on the rest of application. By providing user interface functionality, it involves forms and any browser-based interactions.
- **Business layer:** The business layer implements the business functionality of the application providing the model of the system based on its criteria and functionality. All error handling can be implemented as well as the user interface boundaries in this layer. For instance, there could be a criterion that mandates the user to use a password with a minimum number of 8 characters to strengthen the security of the application.

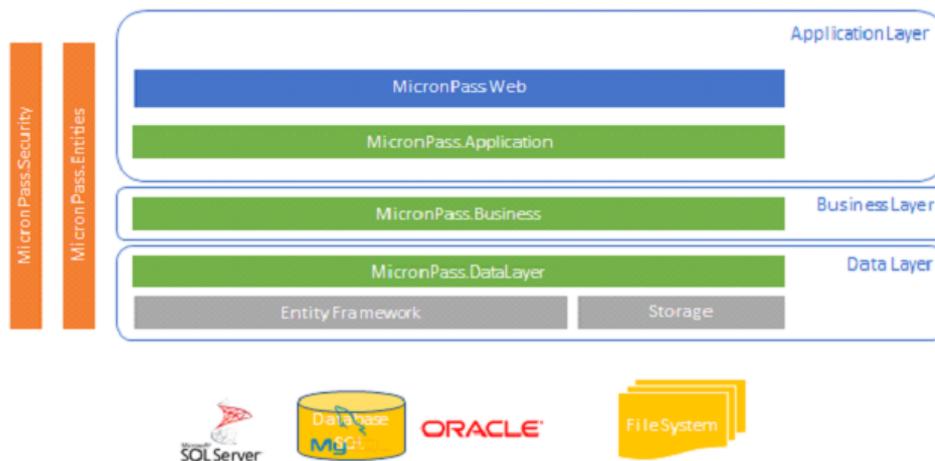


Figure 3.2: New system architectural view

- **Data layer:** The data layer provides any request and response to the server. The server can be located in a local database or a remote database located outside of internal network or in some cases it can be an API (application programming interface) that used as an interface to provide the necessary data through some network protocols. The application layer communicates with the business layer and the business layer interact with the data layer so that the application is loosely coupled, to make the dependency between the layers low. Although it provides a good management of complex requests that involve data manipulation, in order to make the application more fast and efficient the interaction to this layer should be minimum as possible because the effect that is going to be applied on the user interface will take more time since its architectural location is far from the user interface. [2]

As fig 3.2 depicts all the service components are done in the application layer. With this implementation of the new system we get the following advantages.

- We will be able to make any change to the application service without touching the business and the data layer. And its also possible to make an upgrade work on any layer after the completion of development.
- Mappings between the conceptual model and the storage-specific schema can change without changing the application code.
- Developers can work with a consistent application object model that can be mapped to various storage schema, possibly implemented in different database management systems. There are so many advantages that can be gained with such structural change regarding the efficiency of the system but also the development process.

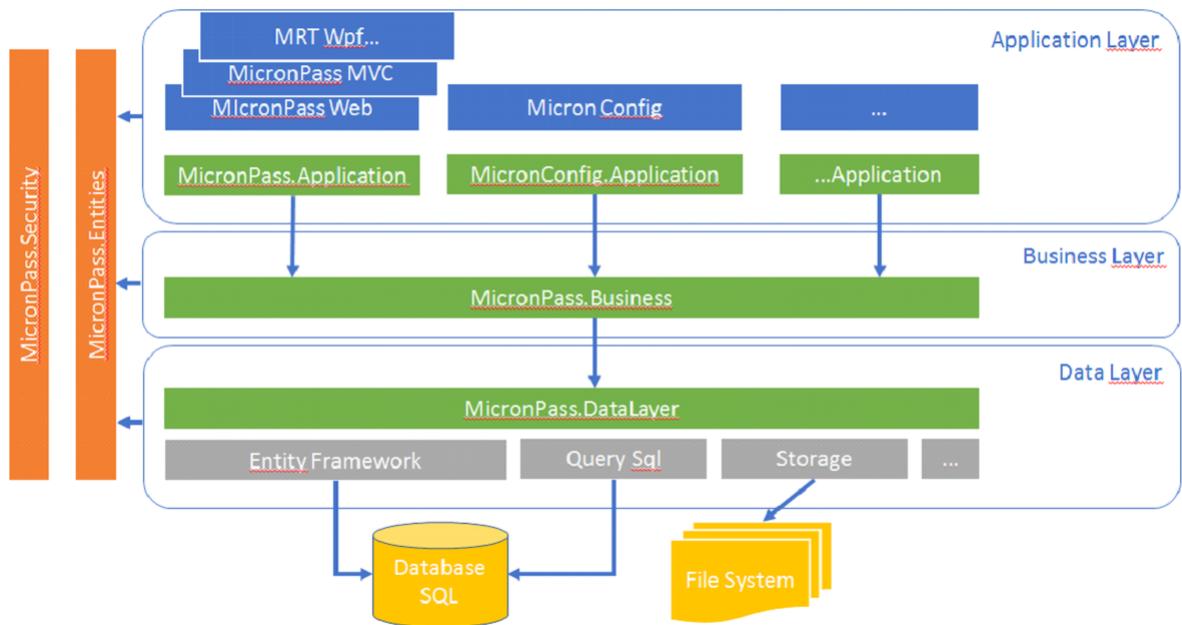


Figure 3.3: New system architectural view with multiple frameworks

The other additional feature of the new system is to allow using multiple application frameworks like `MicronPassWeb`, `Micronpass MVC`, `MRT-wpf` or other new types of frameworks. Additionally, in all level of architecture layer, we have security coverage (`MicronPass-Security`). And the entire layer can access the `MicronPass-Entity`.

The flow of accessibility between the layers is hierarchical. The application layer accesses only the business layer and the business layer accesses the data layer. And with the introduction of Entity frame, the data layer will use the entity framework to access the Sql database or the file system.

The application layer won't access the data layer without the interaction of the business layer in the middle. Also, each layer will have `MicronPass-Entity` and `MicronPass-Security`. This will make each layer to implement security which makes the whole system more secure.

The layers also access `MicronPass-Common`. The `MicronPass-Common` is a collection of common functionality of the whole system. Because of this whenever the layers want to access some more frequent tasks, they can just access `MicronPass-Common`. This will improve the simplicity of the system as well as the performance. For example, if the developer wants to make a change on some basic functionality of the system (which is common to all layers) there will not be a need to make in each layer. In such a way the developer will only change the `MicronPass-Common` so that the changes could be applied to the whole system.

The Entity Framework enables developers to work with data in the form of domain-specific objects and properties, such as customers and customer addresses, without having to concern themselves with the underlying database tables and columns where this data is stored. With the Entity Framework, developers can work at a higher level of abstraction when they deal with data and can create and maintain data-oriented applications with less code than in traditional applications. Saying this about Entity framework in our case there are times that we don't use this. The reason is that there are some cases that need a more complex query and in such cases it's hard to use only entity framework that means there are times that needs to implement sql query.

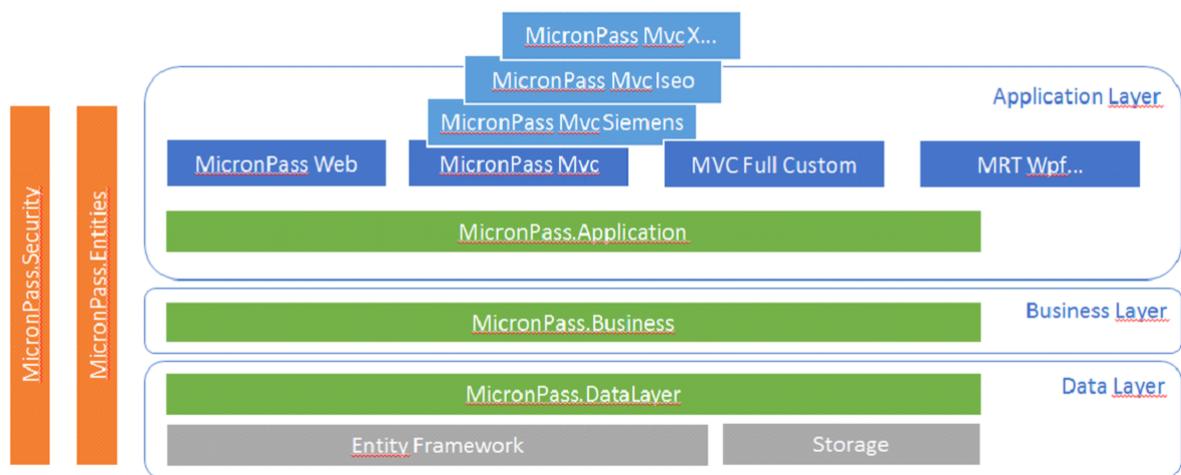


Figure 3.4: New system architectural with multiple MVC application

The other feature of the new system is that it's possible to use multiple types of Mvc on each application layer. There can be a use of `MicronPass-mvcSiemens`, `MicronPass-mvcIseom`, or other types of `MicronPass-mvc` on top of the application layer.

On the database side, there is a possibility to use SQL server, MySQL, Oracle or even file system.

The system will be centralized so that accessing the database will be through a centralized gateway.

3.2 ASP.NET MVC entity framework

The basic developing tool that is used to develop the new system is ASP.NET MVC entity framework. One of the essential parts of an ASP.NET MVC is the architectural design of the model- view-controller (MVC) pattern. It is designed to guarantee the separation between the business logic and the visual appearance of the application.

The model is designed to manage the business logic and implemented in a business layer of MVC application. The controller is used to provide the interaction between the model and the view. It also manages all request from the interface to the server and response from the server to the interface because all http request response is managed and controlled by the controller. The view is all about the front-end of an application. It is responsible for whatever is displayed on the user screen. Separation of concern is applied to make the model, the view, and the controller not to know to each other. Thus the model doesn't know anything about the view, and the view doesn't know anything about the controller.

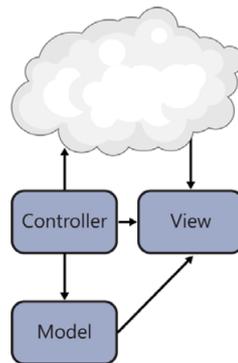


Figure 3.5: Model-View-Controller

- **Model:** The model is designed to manage the business logic. A model object will interact with the data access and based on those data it will do the business logic. Unlike controller and View, the role of the model does not implement any particular interface. The controller is the one who instantiates the model. after the instantiating the controller provides the model to the view so that the view can display based on the characteristics that have been set in the model.

A model role is not always necessary to create a working ASP.NET MVC application. There are times that we can build some part of a solution without the need of model role.

- **Controller:** In ASP.Net MVC, Controllers are the primary handler of the interaction from a user. Controllers role can be error handling when the user input is wrong, getting the values in the model and then executing them based on the application logic, handling the incoming request, and handling user input interaction. A controller calls the model to get the business logic if it exists

and then calls the view to create and render the output Hypertext markup language (HTML).¹

- **View:** The first interaction with an application is through the view. The controller gives a model that is going to be referenced by the view. we can consider some types of view pages. The two most significant ones are the partial page and the layout or the master page.

The partial page is part of interface that display without the HEAD ² and HTML tag. It is used to set some part of interface with a specific task that might be used for some cases. The partial view can not be used to display a layout page

The master or the layout page is essential to share the same layout to multiple pages. Most part of its page is wrapped in HTML so that it can be understood by the web browser.

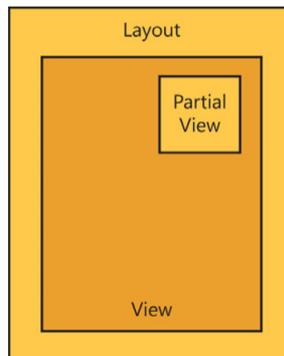


Figure 3.6: View

3.2.1 Entity framework

The entity framework enables accessing the database with a very easy and efficient way with small lines of code. It offers so much options that make the developers to choose based on some situations. The entity framework allows to support the Code First, Model First, and Database First design approaches.

The Model first approach is used mostly when designing the database and designing the object model at the same time. Thus the design that is going to be done on the model object will also be the design schema of the database which then make to generate the sql for the database exactly based on the designed model. Its greatly advantageous when developing a new project to develop without even the existence of database.

The code first approach is used when the code development process is done before

¹ HTML is a markup language that use cascading style shit or JavaScript to create the web page interface

² The head is part of html document that is not displayed on the interface. it includes data like title, links and other additional information of a web page

the creation of the database so that the Code First generator builds the database from those code. By using this method a developer can design the object structure in code that suits the application and generate the database from that design.

In our case since we are using a database that is already created, Database first method is the ultimate choice. The Database first approach enables to continue using an existing structure with no impact on the database.

When accessing the database there are two primary access options. The first one is using object relational mapper (O/RM). ORM is a system that helps in the conversion process of data within a relational database management system(RDBMS) and the object model that is necessary to use in the development framework. What the ORM does is it creates SQL statement from object oriented code or taking the SQL statement and converting to object oriented code to make the data accessing process smooth.

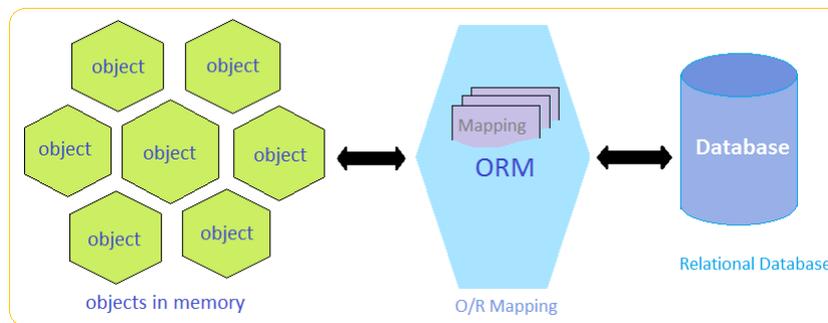


Figure 3.7: Repository

The second option is to write row components that manages the interactions with the database. This needs to manage every conversation between the object model and the database. This seems to be tiresome but there are some situations that needs to use this approach. for example when there is a need to use non relational database like NoSQL ³

³ NoSQL is to mean "Not only sql" which is a mechanism mostly used for large data management and manipulation that uses different mechanism than the normal tabular relations used in normal database.

Chapter 4

New technologies

4.1 Introduction

The biggest change that has emerged in the new system is using new technologies. Since the development of the old system, there have been so many technology changes in the designing environment of a user interface.

In the process of any web-based application, there are two big main parts. The part at the front-end and the part at the back-end. The front-end part deals with whatever the user sees on the user interface: the graphics, the color choice, the size of each displayed components, the density and opacity and all things that are related to the user interface are programmed by the front-end programming part.

Rather all the parts of the developing process that are not front-end and not related to the interface are back-end. The Back-end part deals with what happens to the business logic that happens after the user clicks a button, submit a form and all other kinds stuff concerning with a database.

The issue with the development of a front-end is that the technologies that are used to develop are always on change. As a reason, there are so many new technology tools that have been created since the development of the old system. Because of this reason many new technology tools and concepts are used in the new application.

Some of the technologies that are added to the new system are Ajax, JQuery, JavaScript, knockout, and many other small tools and libraries that are useful for the improvement of the user interface.

The effects of using this tools in the new system will be explained in detail in chapter 5 with a comparison to the old one by taking some interfaces as an example.

Ajax:

One of the most used new technologies that are used in the new version of this project is Ajax. Ajax is a client-side script which is just a source code that is executed on the client's browser instead of the web-server, and it allows for the creation of faster and more responsive web applications.

This client-side script enables the communication from and to a server/database without the need for a post-back which is a submission of a page to the server for processing or a complete page refresh. In another word, Ajax is the method of exchanging data with a server and updating parts of a web page without reloading the entire page. Ajax itself is mostly a generic term for various JavaScript that is used to connect to a web server dynamically without necessarily loading multiple pages. Some of the main benefits of using Ajax in a web application are described below.

- **Callbacks:** Ajax is used to perform a callback; making a quick round trip to and from the server to retrieve and/or save data without posting the entire page back to the server. By not performing a full postback and sending all form data to the server, network utilization is minimized and quicker operations occur. In sites and locations with restricted bandwidth, this can greatly improve network performance. Most of the time, the data being sent to and from the server is minimal. By using callbacks, the server is not required to process all form elements. Sending only the necessary data, there is limited processing on the server. There is no need to process all form elements: send images back to the client, or send a full page back to the client.
- **Making Asynchronous Calls:** Ajax allows you to make asynchronous calls to a web server. Usually without Ajax, when a user makes a request to a server, for example, asking a server for a calculation of a mathematical equation, the user have to wait until the server performs the operation and return the whole content of the page with the final result and in the meanwhile the client has to wait for the response in a waiting blocked state. To overcome this problem, Ajax avoids the client browser to wait for all data to arrive before allowing the user to act once more. This also has a vital effect on the performance and speed of the project.

- User-Friendly:
Because a page postback is being eliminated, Ajax-enabled applications will always be more responsive, faster and more user-friendly.
- Increased Speed: The main purpose of Ajax is to improve the speed, performance, and usability of a web application. A great example of Ajax is the movie rating feature on Netflix. The user rates a movie and their personal rating for that movie will be saved to their database without waiting for the page to refresh or reload. These movie ratings are being saved to their database without posting the entire page back to the server. There will not be any reloading of the whole page.

Ajax should be used anywhere in a web application where a small amount of information could be saved or retrieved from the server without posting back the entire pages. Another good example of this is data validation on save actions or to change the values in a drop down list-box based on other inputs. For example, if we have a list of states and list of university and colleges that each state has, When the user selects a state, the college list box will repopulate with only colleges and universities in that state. This operation requires accessing the database and making some filter operation on the list of received list of data. With Ajax, all these operations will be done without the need of reloading the page.

Other features include text hints and auto-complete text boxes. When a user type a couple of letters and a list of all values that start with those letters will appear below as a suggestion. To enable this feature a callback is made to a web service that will retrieve all values that begin with these characters. This is a very important feature that would be impossible without Ajax and is also part of the Ajax Control Toolkit. To mention a very usual example, when we search “Microtel” on Google, some bunch of list appears as a suggestion.



Figure 4.1: Illustration of Ajax suggestion method used by google

4.2 Practical explanation

Lets' do a simple ASP.Net solution which takes two numbers and perform addition. And let's compare doing this solution with ordinary ASP.Net and ASP.Net with Ajax solution. In such a way we can compare the performance, the efficiency, and the whole flexibility difference.

4.2.1 Code comparison

Usual Asp index html page

In creating any web page using Asp.net MVC, (will be discussed more in the coming section of this Chapter) there are two necessary c-sharp files that are mandatory for the creation process. These are the html page file with an extension of *.cshtml and the controller file with the file an extension of *.cs. There is also a model page file with extension of *.cs which is responsible for additional characteristics of the view but it is not always mandatory.

With the ordinary ASP In the view index, we have two textbox and a submit button. The ViewData brings whatever that is putt in the ViewData of the controller with the id of “result”. In this case, the ViewData contains the summation result of the two numbers that have been calculated in the controller.

```

1
2  @{
3      Layout = null;
4  }
5
6  <!DOCTYPE html>
7
8  <html>
9  <head>
10     <meta name="viewport" content="width=device-width" />
11     <title>Index</title>
12 </head>
13 <body>
14     <div>
15         @using (Html.BeginForm("Index", "Addition", FormMethod.Post))
16         {
17             <span>Enter 1st number :</span>@Html.TextBox("firstNum") <br /><br />
18             <span>Enter 2st number :</span>@Html.TextBox("secondNum") <br /> <br />
19             <button type="submit">Result </button>
20             <span>@ViewData["result"]</span>
21
22
23         }
24     </div>
25 </body>
26 </html>
27

```

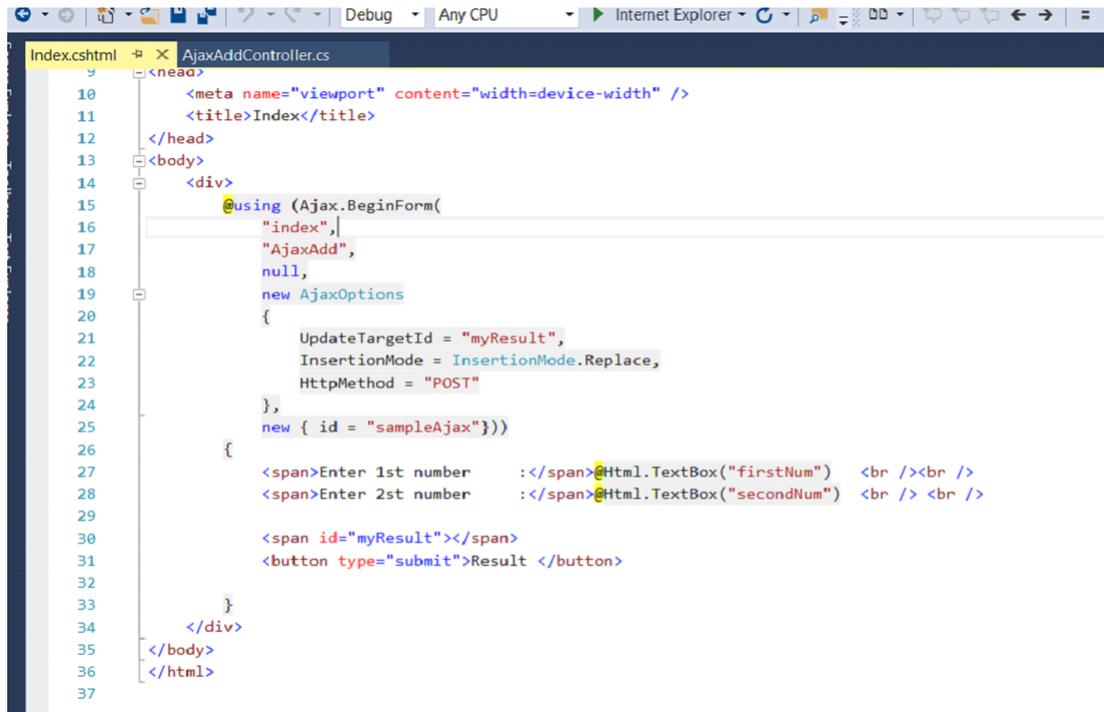
Figure 4.2: Simple addition of two numbers index code

Ajax index page

In the case of Ajax only the result will be posted after the submission of the two numbers.

In the function set of AjaxOption (line 19 in the picture below), the Ajax code will only update the target id “myResult” which holds the result of the two number. This means when a submit button is clicked only the result will be updated.

As a reason unlike the index above, the whole page will not be reloaded when submitting. We will see this more clearly when we inspect the network in a browser.

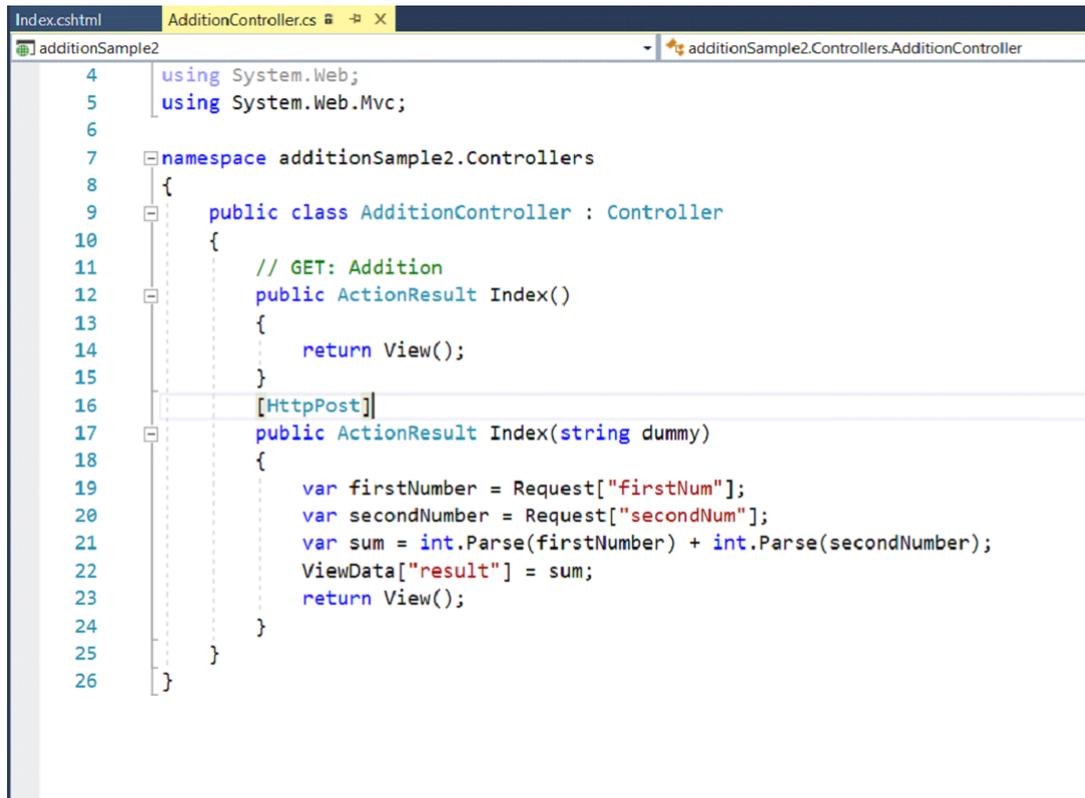


```
10 <meta name="viewport" content="width=device-width" />
11 <title>Index</title>
12 </head>
13 <body>
14 <div>
15     @using (Ajax.BeginForm(
16         "index",|
17         "AjaxAdd",
18         null,
19         new AjaxOptions
20         {
21             UpdateTargetId = "myResult",
22             InsertionMode = InsertionMode.Replace,
23             HttpMethod = "POST"
24         },
25         new { id = "sampleAjax"}))
26     {
27         <span>Enter 1st number :</span>@Html.TextBox("firstNum") <br /><br />
28         <span>Enter 2st number :</span>@Html.TextBox("secondNum") <br /> <br />
29
30         <span id="myResult"></span>
31         <button type="submit">Result </button>
32     }
33 </div>
34 </body>
35 </html>
36
37
```

Figure 4.3: Simple addition of two numbers Ajax index code

Usual ASP control page

First the controller takes the two texts (“firstNum” and “secondNum”) and put the result in Var sum. Then the ViewData will put the sum result in id “result” ViewData (the id should be the same as the id name in the view). Therefore the view will return the result in the ViewData.



```
4 using System.Web;
5 using System.Web.Mvc;
6
7 namespace additionSample2.Controllers
8 {
9     public class AdditionController : Controller
10    {
11        // GET: Addition
12        public ActionResult Index()
13        {
14            return View();
15        }
16        [HttpPost]
17        public ActionResult Index(string dummy)
18        {
19            var firstNumber = Request["firstNum"];
20            var secondNumber = Request["secondNum"];
21            var sum = int.Parse(firstNumber) + int.Parse(secondNumber);
22            ViewData["result"] = sum;
23            return View();
24        }
25    }
26 }
```

Figure 4.4: Controller page of the normal ASP.

Ajax controller page

Here we just take the first and the second number as an argument, put the result in var result and return the result as a string. As we can notice there is no need of use of ViewData and returning of view. The return can be Jason, xml or anything that we want.

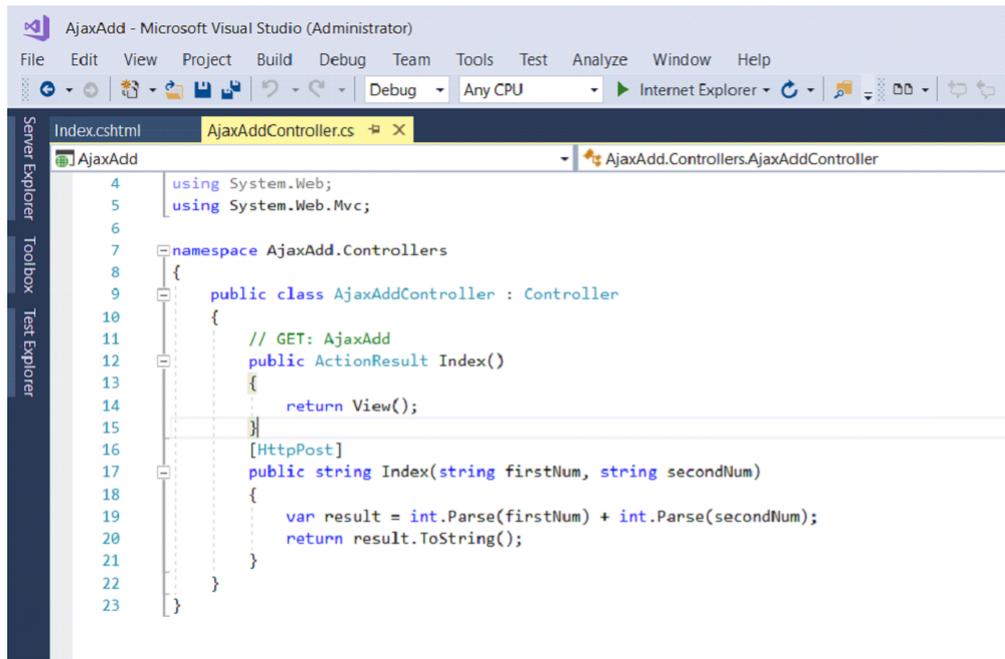
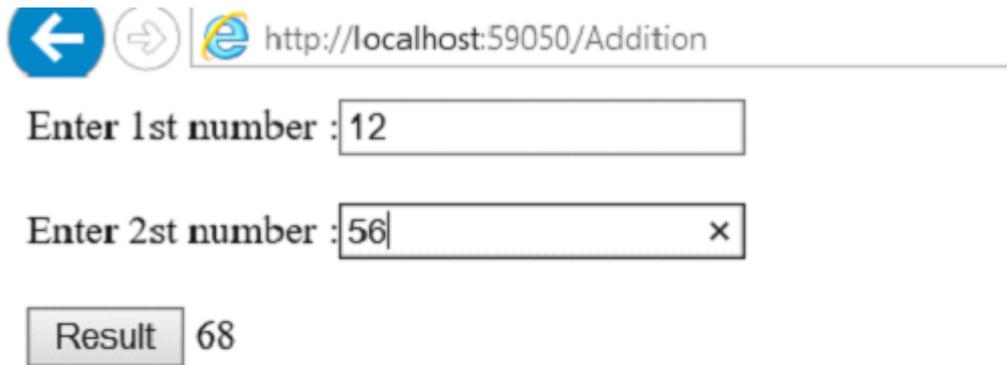


Figure 4.5: Controller page of the Ajax Controller.

The output of the above codes on a browser looks like as follows. The figure below shows the interface of the running version of the above codes. It just takes two numbers in this case 12 and 56 and displays the result 68.

There is no difference between the interface between the Ajax and the normal one. But the unseen difference is huge.



← ↻ | e http://localhost:59050/Addition

Enter 1st number : 12

Enter 2st number : 56 ×

Result 68

Figure 4.6: The Interface of addition of two numbers

Inspection of the normal ASP.

When a submission of the two numbers is executed which means performing a post method, with the normal ASP the whole page is loaded in the response body but the only line that is needed was line 16 (in the figure below) which has the result.

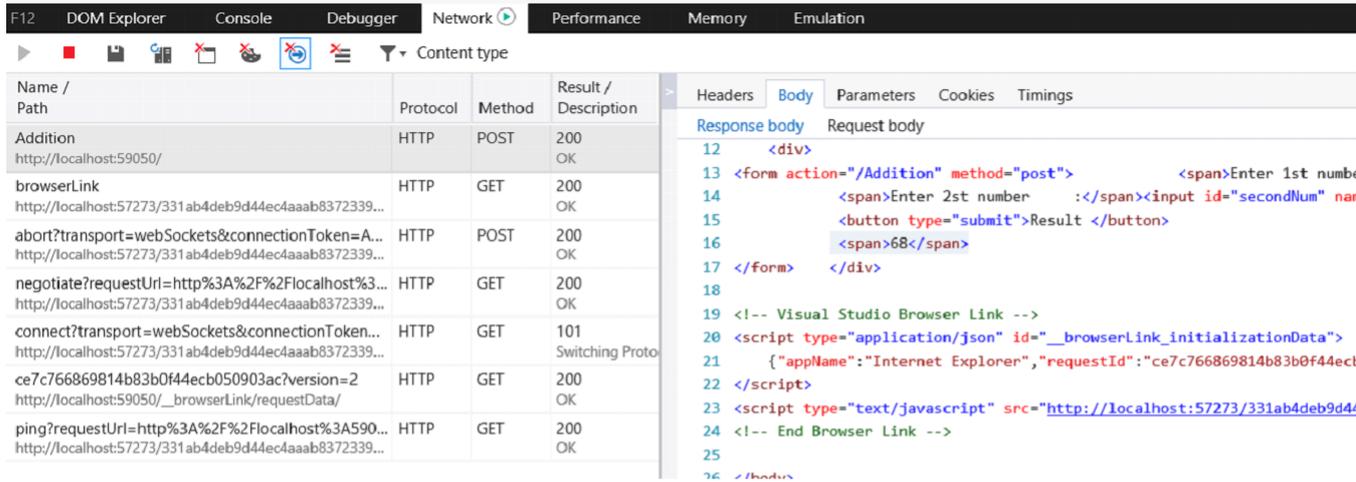


Figure 4.7: Inspection Ajax response body

Inspection of Ajax result.

In this case, when a post method is performed, only the result is loaded in the response body. This clearly affects the speed the performance and the efficiency of our solution. This is the whole concept of using Ajax.

Ajax can use a standard web browser, such as Firefox, Internet Explorer or Safari, as their only user interface. It doesn't force the user to wait for the web server every time the user clicks a button. This is what "asynchronous" means. For instance, Gmail fetches new email messages in the background ("asynchronously") without forcing the user to wait. This makes an AJAX application respond much more like a "real" application on the user's computer, such as Microsoft Outlook.

The Ajax engine works within the Web browser (through JavaScript and the DOM) to render the Web application and handle any requests that the customer might have of the Web server.

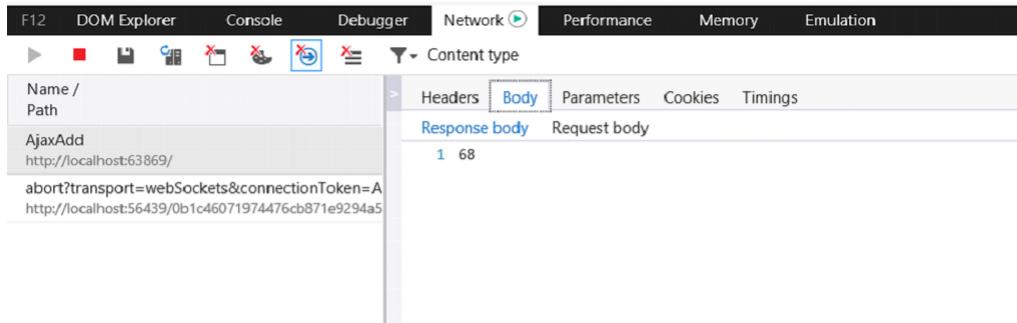


Figure 4.8: Inspection Ajax response body

The beauty of it is that because the Ajax engine is handling the requests, it can hold most information in the engine itself while allowing the interaction with the application and the customer to happen as asynchronously and independently of any interaction with the server.

with Ajax, the JavaScript that is loaded when the page loads handles most of the basic tasks such as data validation and manipulation, as well as display rendering the Ajax engine handle without a trip to the server.

At the same time that it is making display changes for the customer, it is sending data back and forth to the server. But the data transfer is not dependent upon actions of the customer.

4.3 Knockout.js

Knockout.js is a minimalist JavaScript framework for web application development. It is a JavaScript library that allows binding HTML elements against any data model. It is primarily used for creating a rich and responsive display as well as dynamic user interface with a clean, underlying data model. Knockout.js works with any type of client-side or server-side technologies.

It usually depends upon the category of the framework being used. MVC and MVVM are widely used in modern web development. Knockout.js is a JavaScript implementation of the MVVM pattern with templates.

Declarative Bindings:-

This allows you to bind the elements of UI to the data model in a simple and convenient way.

When you use JavaScript to manipulate DOM, this may cause broken code if you later change the DOM hierarchy or element IDs. But with declarative bindings even if you change the DOM then all bound elements stay connected.

You can bind data to a DOM by simply including a data-bind attribute to any DOM element.

Dependency Tracking:-

This automatically updates the right parts of your UI whenever your data model changes. This is achieved by the two-way bindings and a special type of variables called observables. You don't worry about adding event handlers and listeners for dependency tracking.

Templating:-

This comes in handy when your application becomes more complex and you need a way to display a rich structure of view model data, thus keeping your code simple.

Knockout can use alternative template engines for its template binding. Knockout has a native, built-in template engine which you can use right away and can be customized. The data and template are executed to determine the resulting markup.

Extensible:-

This implements custom behaviors as new declarative bindings for easy reuse in just a few lines of code. Knockout is also flexible to integrate with other libraries and technologies.

4.3.1 JSON

JSON is a lightweight, text-based data exchange format based on a subset of the literal notation from the JavaScript programming language.

It provides a succinct encoding for application data structures and is typically used in scenarios where a JavaScript implementation is available to one or both of the applications exchanging data, such as in Ajax-style web applications.

Here is how a simple JSON piece of data may look (140 characters):

```
{
  "id": 123,
  "title": "Object Thinking",
  "author": "David West",
  "published": {
    "by": "Microsoft Press",
    "year": 2004
  }
}
```

A similar document would look like this in XML (167 characters):

```
<?xml version="1.0"?>
<book id="123">
  <title>Object Thinking</title>
  <author>David West</author>
  <published>
    <by>Microsoft Press</by>
    <year>2004</year>
  </published>
</book>
```

4.3.2 Sample effects of the new technologies

Since the process for refactoring the old system application is on progress, the new technologies and tools are applied only on some pages for now. One of the pages that have been refactored is the page that implements the time slot.

The time slot page is used for the interface implementation of the time slot profile. By using the new technology and new architecture its enabled to provide efficiency by managing more than 100000 number of entrances.

It has its corresponding view page that does all the accomplishment of displaying the final interface.

The page is used to modify and edit a specific guest template type. Taking its code on a session and making the modification on it.



Figure 4.9: Time slot edit page

This page uses Ajax method for reading data as well as for saving changes to profile templates. It saves a single configuration item in Add or Update mode. It also has a function to perform a move up and a move down, sort, filter with the addition function for deletion of items. The final user interface is shown below.

By clicking the + sign in the above page we can add a specific type of time slot to the guest model template that is being edited.



Figure 4.10: Binding prompt

The popping prompt will give the user the information about the detail that is going to be added and an option to confirm the adding process or the option to cancel

4.3.3 Filtering description

One of the biggest advantages that has been added to the new system is how a user can filter. In the old system, any filtering requires clicking pressing enter after typing what to filter as well as the result will be shown after the reloading of the page.

Unlike in the old system, the new system application has added so man advancements using the new technologies.

for instance, a user can filter based on two or more conditions and the result can be shown while the user is typing.

To following figures can be used to illustrate the different types of filtering that can be used by using the new technologies These forms have many filtering criteria that make the searching method so easy and efficient. In each filtering new technology is used such as Ajax, knockout.js and many more which are described below.

The screenshot shows a web application interface with a green header bar. The header contains the text "Versione 7.4.26" and "Profilo di accesso del modello 00000007-Daily visit". Below the header, there is a search form with a text input field containing "002", a dropdown menu for "Tutte le tecnologie", and another dropdown menu for "Varchi e gruppi". Below the search form, there are two text input fields: "Impianti" and "Sedi logiche". To the right of these fields are the links "Aggiorna" and "Indietro". Below the search form, there is a table with the following columns: "Codice", "Descrizione", "Impianto", and "Sede". The table contains one row with the following data: "+ 00000002", "VARCDUE", "EVVE UNO DUE", and "Salca". Below the table, there is a pagination control showing "Vista da 1 a 1 di 1 elementi" and two buttons: "Precedente" and "Successivo". At the bottom of the page, there is a message: "Definizione profilo mancante".

Figure 4.11: Filtering by code and description

The first filter text box is used to filter by code and the description.
The search spelling can be in any position.

Versione 7.4.26 Profilo di accesso del modello 00000007-Daily visit

Filtro: Tutte le tecnologie ▼ Varchi e gruppi ▼

Sedi logiche:

[Aggiorna](#) | [Indietro](#)

Codice	Descrizione	Impianto	Sede
+ 00000002	VARCDUE	EVVE UNO DUE	Salca
+ 00000004	VARCODIO	EVVE SALCA	Salca

Vista da 1 a 2 di 2 elementi

[Precedente](#) [Successivo](#)

Definizione profilo mancante

Figure 4.12: Filtering by Plant

We can also filter by plant. The good thing here is we can use more than one criteria at once thanks to knockout.js.(further discussed on the next chapter)

Versione 7.4.26 Profilo di accesso del modello 00000007-Daily visit

Filtro: Tutte le tecnologie ▼ Varchi e gruppi ▼

Impianti:

[Aggiorna](#) | [Indietro](#)

Codice	Descrizione	Impianto	Sede
+ 00000002	VARCDUE	EVVE UNO DUE	Salca
+ 00000004	VARCODIO	EVVE SALCA	Salca
+ 00000003	VARCTRE	EVVE DDUE	Salca
+ 00000001	VARCUNO	EVVE UNO UNO	Salca

Vista da 1 a 4 di 4 elementi

[Precedente](#) [Successivo](#)

Definizione profilo mancante

Figure 4.13: Filtering by logical seat

The same type of filtering is applied for the logical seat.

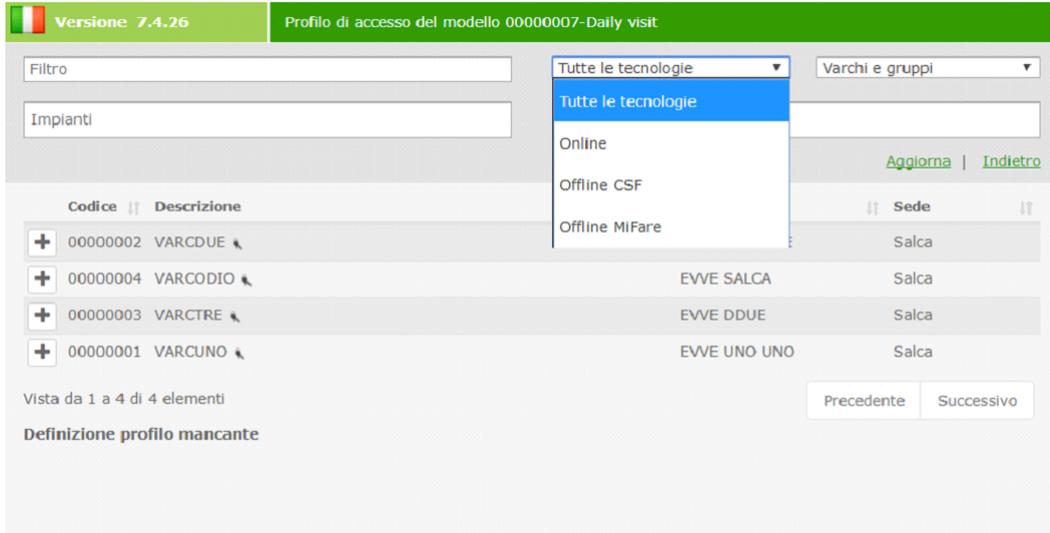


Figure 4.14: Filtering by Technology

The filtering by technology has some option that allows the user to choose. The user can choose all technology used, online technologies, offline with CSF, or offline with done by the user itself

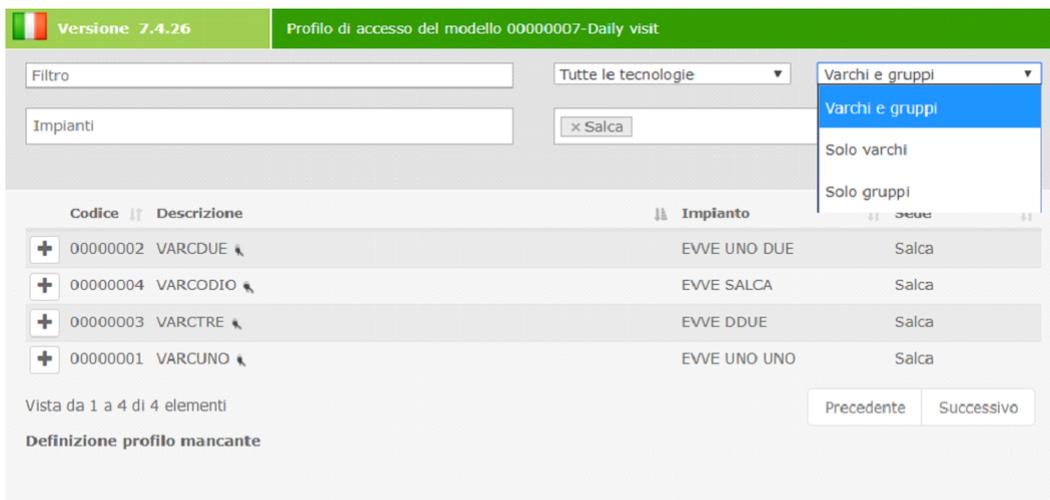


Figure 4.15: Filtering by Gate or group

Finally, we get an option to filter by gates and group, only by gates or By using the only group

Chapter 5

Comparison of the two systems

5.1 Time slot

This chapter is dedicated to making a comparison briefly between the old and the new system. The old system application uses MVC EF or LINQ methods. And also there is no common layer that used as a bridge between the databases. It directly accesses the database with a text query (normal sql procedure).

This kind of technology and method of development lead to many drawbacks and it was also so difficult to develop for the programmers. In addition, the flexibility of the system is so backward. For example any change on the interface or on any layer impacts the whole system. The other issue is performance. With the new technology, the size of the solution is much less because many redundant functions have been eliminated since common architecture is implemented and some new methods are used. This results to fasten loading of the solution as a reason increasing the performance.

The new application is developed using MVC EF with an implementation of MRTcore which has a great role by providing a shared ground for the whole system and allowing to be used by other subsystems. It does not have any query since it uses the MRTcore. The MRTcore does all the interaction with the database so that there will not be a direct interaction between the upper layers and the database. This methodology and using of the new technology makes the new system to overcome all the drawbacks that have been occurring on the previous version. With the new version, we will have the same functionality with more flexibility, with more performance and with less line of codes.

5.1.1 Name Space Comparison

Assuming that the programming language is Microsoft ASP.Net framework, NameSpace is the Logical group of types or a container of Class, Structures, Interfaces, Enumerations, and Delegates etc, with a common functionality. For example, NameSpace System.IO logically groups input-output related features, NameSpace System.Data.SqlClient is the logical group of ado.net Connectivity with Sql server related features. In Object Oriented world, programmers frequently class name which can lead to conflict error. Qualifying NameSpace with class name can avoid this collision.

Table 5.1: Name Spaces In Common between the two projects

Name spaces	Description
Imports System.Globalizaton	Contains classes that define culture-related information, including language, country/region, calendars in use, format patterns for dates, currency, numbers, and sort order for strings. These classes are useful for writing globalized (internationalized) applications. Classes such as StringInfo and TextInfo provide advanced globalization functionalities, including surrogate support and text element processing.
Imports System.Web.HttpUtility	Provides methods for encoding and decoding URLs when processing Web requests. This class cannot be inherited.
Imports System.Collections.Generic	Contains interfaces and classes that define generic collections, which allow users to create strongly typed collections that provide better type safety and performance than non-generic strongly typed collections.
Imports btLib.user Imports btLib.Licensing	Both name spaces are used to use some functionalities of btLib library
Imports mPassW.Legacy	For the use of Legacy subsystem solution.

There are some Name spaces that are only used on the old application. They are not used in the new application.

Table 5.2: Name Spaces only on the old application

Name spaces	Description
Imports System.Threading	Creates and controls a thread, sets its priority, and gets its status. To have several threads to be able each thread run independently without affecting any other threads or user process. .
Imports btLib.DBU Imports btLib.BTDB	Both name spaces are used to use some functionalities of btLib library.
Imports mPassW.Legacy	For the use of Legacy subsystem solution.

There are also some Name spaces that are only used on the new ongoing application. They are brought here because of the additional frameworks used like MVC entity framework and libraries.

Table 5.3: Name Spaces only on the new application

Name spaces	Description
Imports System.Linq	This name space provides classes and interfaces that support queries that use Language-Integrated Query (LINQ). LINQ extension method overrides that offer greater efficiency for ImmutableArray. (cannot be changed once it is created.) than the standard LINQ methods.
Imports MrtCore.Entities Imports MrtCore.Application Imports MrtCore.Common.Enums Imports MrtCore.Common.Exceptions Imports MrtCore.Common.Extensions	These name spaces are used to access the MrtCore solution that is used as a bridge between the main solution and the database. It enables a hierarchical structure for the development process allowing to be used by other solutions if any.
Imports mPassW.Legacy Imports mPassW.Mvc.Controllers Imports mPassW.Mvc.Exceptions Imports mPassW.Mvc.Models.TimeSlot	All the name spaces mentioned above are rooted from the solution mPassW.Mvc which is responsible for all functionalities that uses the MVC. Which is, In addition to managing complexity, the MVC pattern makes it easier to test applications than it is to test a Web Forms-based ASP.NET Web application. For example, in a Web Forms-based ASP.NET Web application, a single class is used both to display output and to respond to user input.

5.1.2 Code comparison

The number of code lines decreases in the new system since there is no query operation. More importantly the new solution uses the MRTcore. This means all operation needed to interact with the database will be done by MRTcore. As the

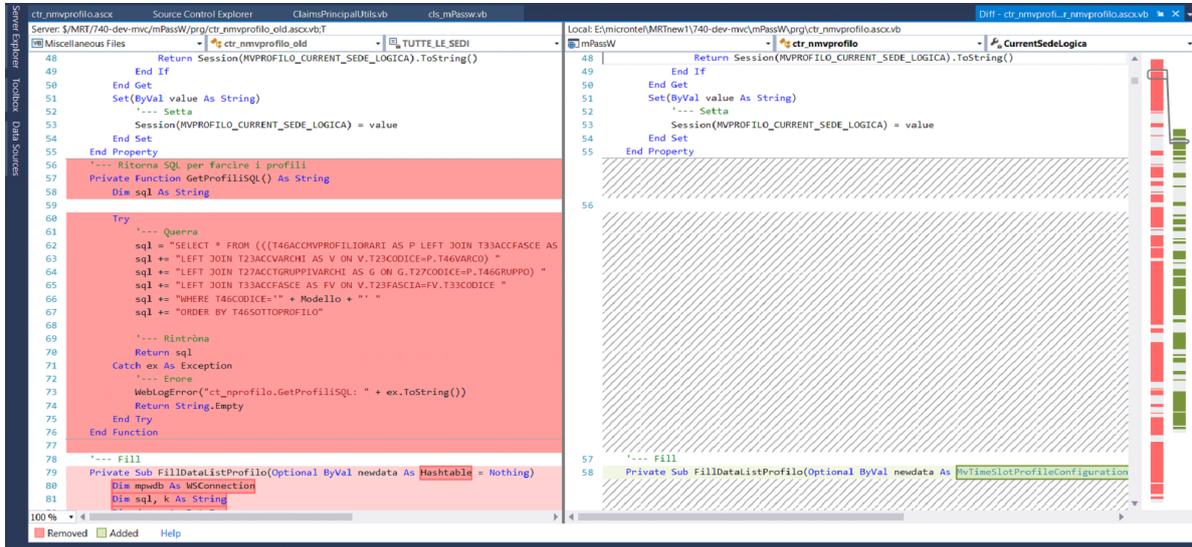


Figure 5.1: Sample Comparison when accessing the database

figure depicts the old version of the solution is full of sql query whereas the new version doesn't have any. The reason behind is that the new version uses the MRTcore which is used for all query operation. But this doesn't mean there is a change of code in the user interface. There is no change between the old and the new version regarding user interface.

The MRTcore uses linq (language integrated query) method for most of the query operation. LINQ to SQL allow us to query and modify SQL Server database by using LINQ syntax. Entity framework is a great ORM shipped by Microsoft which allows us to query and modify RDBMS like SQL Server, Oracle, DB2 and MySQL etc. by using LINQ syntax. LINQ offers a compact, expressive, and intelligible syntax for manipulating data.

LINQ can be used to query many different types of data, including SQL, XML, and even objects. It is a Microsoft programming model and methodology that gives formal query capabilities into Microsoft .NET- based programming languages (mainly in CSharp and VB.Net). LINQ queries can be written through standard query expression or through Lambda expressions. Query expression syntax:

```
var items = from item in Items where item.Price > 10 select item;  
Lambda expression syntax: ( nice and easy to use)  
var items = Items.Where(c => c.Price > 10).Select(c => c);
```

Queries written with LINQ are cleaner and typesafe as the queries errors are type checked at compile time. It also can be used against different data types and it isn't limited only to relational databases, you can also use it against XML or regular objects.

With LINQ one can able to query not just tables of a database but also able to query on XML and text files. Classes and properties are automatically created in LINQ and you can work with these classes instead of querying data directly to database. Relationships in LINQ are automatically appeared to classes, if there are proper foreign keys.

Deployment is easier with LINQ, as in case of deployment, we need to provide an additional script for DB to execute but in case of LINQ, it will compiled into single DLL.

Although LINQ has the advantages mentioned above, it has also some drawbacks.

- Performance is degraded if LINQ queries not written in correct manner.
- If there has been a change in LINQ query, then assembly needs to be recompiled and redeployed.
- LINQ is generic, whereas queries written in database can take full advantage of the complete database features.
- It's much easier to restrict access to tables in database using queries than through LINQ.
- LINQ statements are not precompiled whereas SQL stored procedures are pre-compiled, so stored procedures are faster in performance as compared to traditional LINQ.

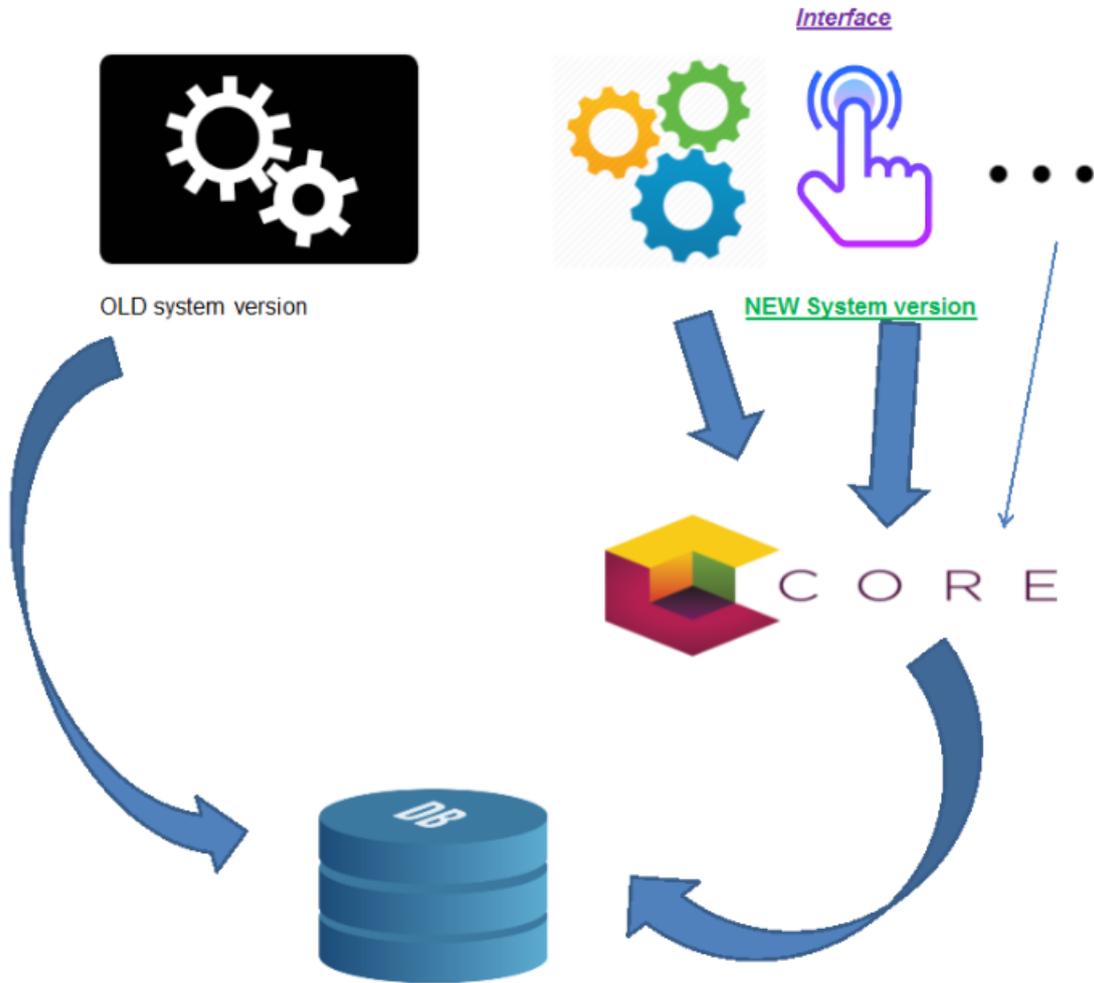


Figure 5.2: Comparison Summary of Old (left side) and new (right side) systems

From the many advantages of the MRTcore, the main important aspect is that it allows many other subsystems to be implemented and modified without any influence on other aspects of the system. For instance, the interface (in the figure above) can make any change by its own without any influence on the business and application layer.

In addition, Multiple User Interfaces can be developed without concerning about each other applying Parallel development. This has an advantage for the developers. First The developers of UI can focus exclusively on the UI screens without bogged down with business logic.

The other advantage is that the developer who is working on the Model can focus exclusively on the business logic implementations, modifications, and updating without concerning the look and feel of the interface. Thus the business logic developers can build the classes, while the UI developers can involve in designing UI screens simultaneously, solving the problem with inter-dependency issues and time conservation.

The result we get from this will be, updating the user interface can be made without slowing down the business logic process subsequently the business logic changes that need the revision of the user interface will be less.

Chapter 6

Unit testing

6.1 Introduction

In computer programming, unit testing is a software testing method by which individual units of source code are tested to determine whether they are fit for use. A unit is the smallest possible testable software component. Usually, it performs a single cohesive function. A unit is small, so it is easier to design, execute, record, and analyze test results for than larger chunks of code are.

Defects revealed by a unit test are easy to locate and relatively easy to repair. The goal of unit testing is to segregate each part of the program and test that the individual parts are working correctly. It isolates the smallest piece of testable software from the remainder of the code and determines whether it behaves exactly as you expect.

Unit testing has proven its value in that a large percentage of defects are identified during its use. It allows automation of the testing process, reduces difficulties of discovering errors contained in more complex pieces of the application, and enhances test coverage because attention is given to each unit.

6.1.1 Advantages of unit testing

[3] Unit testing provides numerous benefits including finding software bugs early, facilitating change, simplifying integration, providing a source of documentation etc. Some of the main advantages of unit testing are the following.

- **Makes the process Agile:** ¹ One of the main benefits of unit testing is that it makes the coding process more Agile. When you add more and more features to software, you sometimes need to change old design and code. However, changing already-tested code is both risky and costly. If we have unit tests in place, then we can proceed for re-factoring confidently.

Unit testing really goes hand-in-hand with agile programming of all flavors because it builds in tests that allow you to make changes more easily. In other words, unit tests facilitate safe re-factoring.

- **Quality of code:** Unit testing improves the quality of the a code. It identifies every defect that may have come up before code is sent further for integration testing. Writing tests before actual coding makes to think harder about the problem. It exposes the edge cases and makes the code better.
- **Finds Software Bugs Early:** Issues are found at an early stage. Since unit testing is carried out by developers who test individual code before integration, issues can be found very early and can be resolved then and there without impacting the other pieces of the code. This includes both bugs in the programmer's implementation and flaws or missing parts of the specification for the unit.
- **Facilitates Changes and Simplifies Integration:** Unit testing allows the programmer to refactor code or upgrade system libraries at a later date and make sure the module still works correctly.

Unit tests detect changes that may break a design contract. They help with maintaining and changing the code.

Unit testing reduces defects in the newly developed features or reduces bugs when changing the existing functionality.

Unit testing verifies the accuracy of each unit. Afterward, the units are integrated into an application by testing parts of the application via unit testing. Later testing of the application during the integration process is easier due to the verification of the individual units.

¹Agile is a software development process management method that is used to make the development process more flexible with a smooth flow by allowing an iterative progress to be able to gradually increase a specific task of a project

- **Provides documentation:** Unit testing provides documentation of the system. Developers looking to learn what functionality is provided by a unit and how to use it can look at the unit tests to gain a basic understanding of the unit's interface (API).
- **Debugging process:** Unit testing helps simplify the debugging process. If a test fails, then only the latest changes made in the code need to be debugged.
- **Design Reduce cost:** Writing the test first forces you to think through your design and what it must accomplish before you write the code. This not only keeps you focused; it makes you create better designs. Testing a piece of code forces you to define what that code is responsible for. If you can do this easily, that means the code's responsibility is well-defined and therefore that it has high cohesion.
- **Reduce cost:** Since the bugs are found early, unit testing helps reduce the cost of bug fixes. Just imagine the cost of a bug found during the later stages of development, like during system testing or during acceptance testing. Of course, bugs detected earlier are easier to fix because bugs detected later are usually the result of many changes, and you don't really know which one caused the bug.

There are many other benefits and advantages of unit testing which makes it a life saver both for the programmer and also for the whole development environment and system process.

6.2 Sample unit test

In the development of our system we made a unit testing for each transaction of the system to make sure everything goes as it supposed to be before the full compilation and publication.

The testing is on each input text box, formats of the date and time, the drop boxes, the list boxes . . . Testing the text box could be checking if the system accepts empty insertion which might be okay if the field is not mandatory. The other checking could be the length of the character. But the maximum length should be known first because the length for description field and for normal text box is not the same.

Checking the type of the field could be another thing that should be checked. Trying to insert string characters in number fields should not be allowed. For example, the system should not accept inserting a string character on the Age field.

In the checking of the date and time insertion of the date of birth less than some threshold will not be allowed or even checking if the system allows accepting a date that is in the future of the current date (greater than the current date). There is also a checking of the date time picker and the format of the date. For example, checking if it accepts a number greater than 12 in the month field or greater than 31 in the date field or inserting a character string on the date and year section (string could be accepted in the month section) and many more other format checks.

The testing will also include some other types of complex checking's. Sometimes the list field or combo box should include the content of another insertion. This needs a little deep knowledge of needed content. Checking whether the list and the combo box include the needed content is mandatory. There are also other crucial testing like user name and password. The system must be good enough to tackle any kind of pony-trekking and attacking in the user and password. The password should be hashed and encrypted so that even if an attacker or any authorized person finds it, they will not be able to know the correct one.

Beside unit testing there are other types of testing which checks the integrity of the system. The speed, the performance, the graphical texture, the navigation's, the flexibility, the layout, the formation of pictures, the transformation of the forms, the tool bars . . . and many other tiny but significant schemas are tested whether by using the unit testing or manual testing. Fig 6.1 depicts the picture of the new

The screenshot shows the MICRONPASS web interface. The header includes the MICRONTEL logo, the title 'MICRONPASS web', and the company name 'Gestione Accessi Ferrero S.p.A. - Alba (CN)'. The version is 7.3.12 and the current date is Sunday, May 14, 2017. The left sidebar contains a navigation menu with various options like 'Home page', 'Employees', 'External Collaborators', etc. The main content area displays a 'New booking' form with the following fields: First Name, Last Name, Company, Date of birth, Document, Birthplace, Scheduled time (5:51:33 PM), Reason of visit, Visit type (Nessun tipo), and Employee. A dropdown menu is open showing 'No one' with 'Change' and 'Remove' links. The form has 'OK' and 'Cancel' buttons.

Figure 6.1: New booking form

booking form. This form uses for book visiting of guests which makes visiting more flexible, time saver, and flexible.

The guest information will be filled in this form and reservation of visitor will be done. All the guest information should be filled. His First name, Last name, the company name, The date of birth of the guest, some documents like identity card or passport, the guest birthplace, the scheduled time that the guest will visit, reason of the visit, visit type, employee name that he/she will visit; this combo box gives all the employees from the “Employee” record. Hence we should test if the system is bringing the employee records

6.2.1 Unit test result

In Unit test process all fields of a page are tested to check their correctness. For instance we can test first-Name text box by checking if it allows empty insertion or not, by filling the other fields and making the First-Name field empty.

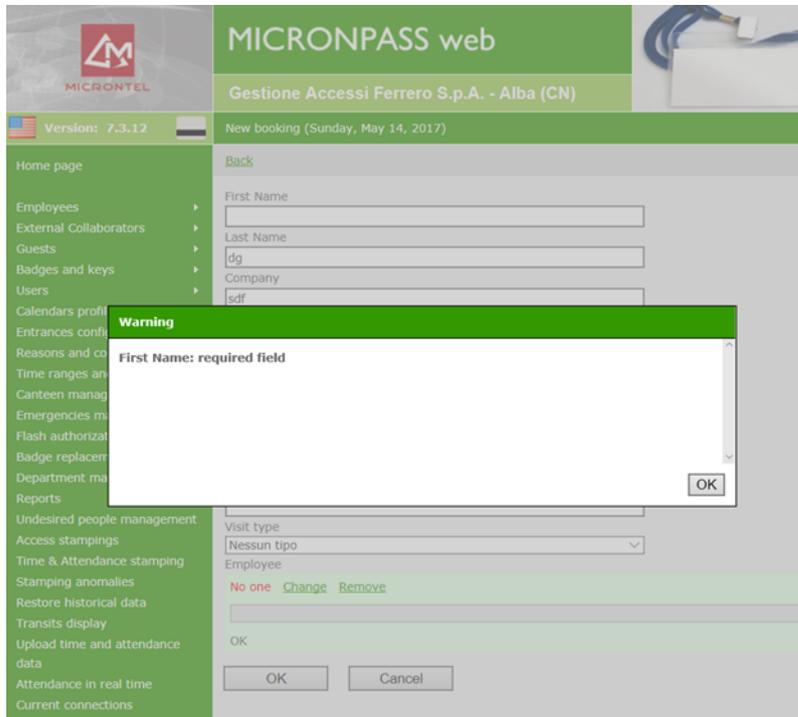


Figure 6.2: Unit test for First-Name

As it is shown in fig 6.2, when we try to insert an empty string in the First-Name field we get the warning popup which mandates the insertion of the First-Name. This is just the graphical interpretation of the unit test. The same unit test is applied to the fields of Last-name, Company, Document, Birth Place and reason of visit.

For the date time, if we try to make a booking with an empty “date of birth”, the system will allow us because “date of birth” is not mandatory. So in this case, on contrary, the unit testing is to check if the system does not block empty “date of birth”.

The unit testing for new booking is summarized in the table below. Since the project aims to improve the an old version of the system we can take a look at both versions

	Test Cases	Results
1	Empty all fields	error Output: "Last Name: required field"
2	Fill first name and empty all other fields	error Output: "Last Name: required field"
3	Fill last name and empty all other fields	error Output: "First Name: required field"
4	Empty company name and fill all other fields	error Output: "Company: required field"
5	Empty Date of birth	Execution success since its not a mandatory field
6	Empty Document	Execution success since its not a mandatory field
7	Empty Birth place	Execution success since its not a mandatory field
8	Empty Schedule time	error Output: "Scheduled time: required field"
9	Empty Reason of visit	error Output : "Motive: mandatory field"
10	Empty Visit type	Execution success since its not a mandatory field
11	Empty Employee	error Output: "Set an employee"
12	Insertion of string (not number) into Schedule time	error Output: "Scheduled time: required field"
13	Insertion of string (not number) into Date of birth	Execution success (it takes it as an empty string)
14	List employee with empty filter string	No change it turns back to "please specify a filter"
15	Remove employee	No change at all
16	Back button	Returns to the previous page
17	Cancel button	Returns to the previous page
18	List employee with first name	Not listing anything
19	List employee with Last name (with no privilege to guest)	Not listing anything
20	List employee with Last name (with a privilege)	Listing the employee record
21	Insertion of future date on the birth of date field	Success which is wrong

Table 6.1: The test case of the old version application

The table below shows the tests and there corresponding result for the ongoing application.

	Test Cases	Results
1	Empty all	Fatal error in writing data, consult the log
2	Fill first name and empty all other	Fatal error in writing data, consult the log
3	Fill last name and empty all other	Fatal error in writing data, consult the log
4	Empty company name and fill all other	Fatal error in writing data, consult the log
5	Empty Date of birth	Fatal error in writing data, consult the log
6	Empty Document	Fatal error in writing data, consult the log
7	Empty Birthplace	Fatal error in writing data, consult the log
8	Empty Schedule time	Fatal error in writing data, consult the log
9	Empty Reason of visit	Fatal error in writing data, consult the log
10	Empty Visit type	No error but the data are not saved
11	Empty Employee	No error but auto-fill s wrong
12	Insertion of string (not number) into Schedule time	Fatal error in writing data, consult the log log
13	Insertion of string (not number) into Date of birth	Fatal error in writing data, consult the log
14	List employee with empty filter string	No change turn back to "please specify a filter"
15	Remove employee	No change at all
16	Back button	Returns to the previous page
17	Cancel button	Returns to the previous page
18	List employee with first name	Not listing anything
19	List employee with Last name (with no privilege to guest)	Not listing anything
20	List employee with Last name (with a privilege)	Listing the employee record
21	Insertion of future date on the birth of date field	Success which is wrong

Table 6.2: The test case of the ongoing version

Chapter 7

Conclusion and Future Work

This dissertation studies about an application system software that is designed for access control service owned by Microntel company. The company has been using an application software to accommodate the access control service built before 15 years. This old application becomes outdated and needed to be refactored. This paper studies about the old application system such as; why it was needed to refactor, what benefit can be acquired by using new software development tools, how was the structure of the old application, what structural changes are done with the new application, and so on. It answers these questions with illustrations by making sample comparison between the two versions.

Development of a web application needs first to design and to study the structural overview of the development process before starting implementation. This process is a base for the outcome result and can take a very significant time. This thesis paper is used to deliver documented information about the applications to get the necessary information in the development of new user interface when re-factoring an already existing web application.

I studied the factors of the two systems such as; an architectural structure of the old system and the new system, about the new technology tools used and their effect in improving the system and why the old system was slow and not efficient by showing sample illustration. I also made a documentation of each project, run some unit tests to know how the new system will be tested and how it will guarantee to deliver the expected outcome.

7.1 Future work

One of the best things about the technology used in the new system is that it accommodates the separation of code (SOC) methodology. This method allows adding any functionality without affecting the rest of the system. Thus any future work can be added at anytime smoothly. Although the application system is a very big project taking at least two years to finish, there will be always a room for improvement.

For a good outcome in the process of developing a web application, it's important to make most of the work in the client side (the browser). When there is a little calling of the server and database, the response time of the system will be very fast because request and response time will be very short. Therefore the future work is intended to implement such method by migrating most of the work to the client side to maximize the efficiency of the system.

The other functionality that could be added to the system is to develop a mobile application for the system. Mobile applications cannot accommodate the whole functionality of the system. As a reason, the task that can be done on mobile will be a small snippet from the whole system. The mobile application will play a very vital role especially when the user wants to use the system while they are in remote areas.

7.2 Conclusion

The old system was not efficient in many ways. For example, when there is any change on a page even a simple small one like selecting from a drop-down option, the whole page will reload and this makes to refresh and reload every file and item in that page. There have been many such kinds of drawbacks in the old system that made it be slow, inefficient, not flexible and not user-friendly.

The problem with the improvement of any system is losing resources that have been exerted to do the old work. The programming languages and frameworks that have been used to develop allows integrating the new application with the old application without losing resources by implementing gradual development method. The new system re-factored the old one to improve a web application system without any negative effect on the old system by implementing some new architectural methodologies to work hand in hand with the old one.

The new web application system will improve the old system and overcome most of its drawbacks to make it more fast, efficient, flexible and refine the looks and feel of user experience.

Bibliography

- [1] Microsoft. *Microsoft documentation*. 1984. URL: <https://www.microsoft.com/en-us/>.
- [2] Microsoft. *Microsoft documentation*. 1984. URL: <https://www.microsoft.com/en-us/>.
- [3] unit Tesing. *Unit Testing*. 1984. URL: <https://docs.microsoft.com/en-us/visualstudio/test/unit-test-basics>.