BABEŞ-BOLYAI UNIVERSITY CLUJ-NAPOCA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

SPECIALIZATION COMPUTER SCIENCE IN GERMAN

DIPLOMA THESIS

My vehichle management

Supervisor

Lecturer Diana Cristea, PhD

Author

Cîrstea Ștefan-Daniel

2022

Contents

Introduction 4
Scope and motivation 5
Web applications 6
Server-side technologies 7
Relational Database 8
Client-side technologies 9
Optical Character Recognition (OCR) 11
Application 12
Client-side structure 12
Frontend key features 12
.Net project structure 12
AI Used principles 12
Application features 12
Database structure 12
Application tutorial 12
References 13

Introduction

Scope and motivation

Today, there are more applications that help people remember things, and many ways to store data about your daily activities. But some deadlines are easy to be forgotten especially when it takes more than one or two months to take place. Because of that, I had chosen to create an application which scope is to help people manage all their vehicle information.

Managing vehicles using mobile or web-based applications is not a new idea and it has been done before, an example of this kind is MOVCAR [1]. Because I know that time is very important in our days, anyone have more to do every day and because I am a driver, I like driving and I know I will always be a driver, my application is looking to help people manage their vehicles. Moreover, is looking to help them by extracting data from their documents.

There are more other applications that remind people about their periodical mandatory activities or documents that need to be done in order to be safe as a driver. In Table 1 shows a comparison between some of this apps and the app developed by me.

Application / Features	Deadline reminder	Fleet management	Fuel Management	Data extraction	Web solution	Service management	Cloud solution
MvManagement	X	X	X	X	X	X	X
MOVCAR	X	X	X		X		X
MyCar	X						X
Car Alert	X						

Table Table 1 - Applications comparison

Web applications

Web applications are developed software that runs on a web server and can be remotely accessed from other devices through a browser interface. This kind of applications are designed to for a variety of scopes from online calculator to web mailing applications, e-commerce shops and even more complex applications. It is usually built from three very important and decoupled parts, frontend or client-side, backend or server-side and database. Client-side represents all the graphical interfaces, directly, what the user sees. Backend is the part in charge of creating the login of the application and the mediator between the frontend and the database, basically it scope is processing data and sending it to the client-side. Database part is the one which is storing the data and is responsible of the correctness of it.

Web applications to not need to be downloaded, in order to run they need a web server, application server host the logic and a database. Web servers are responsible for hosting the client-side of the application and for managing the request that comes from the client. Application server completes a requested the task and then returns data to the web server and database is basically used for storing any information needed.

Communication between web application components is also a very important part in application development process. One of the most used interfaces for transferring data bidirectional between software applications are REST APIs (Application Programming Interface). APIs are designed as a bridge that let data travel between two applications. Basically, this is how the backend and frontend most commonly communicate over the HTTP protocol. The HTTP protocol is the most used protocol for transferring information between a Web Browser and a web server, and it represent a text protocol. The entire communication process is illustrated in Figure 1 - Application Rest API communication.

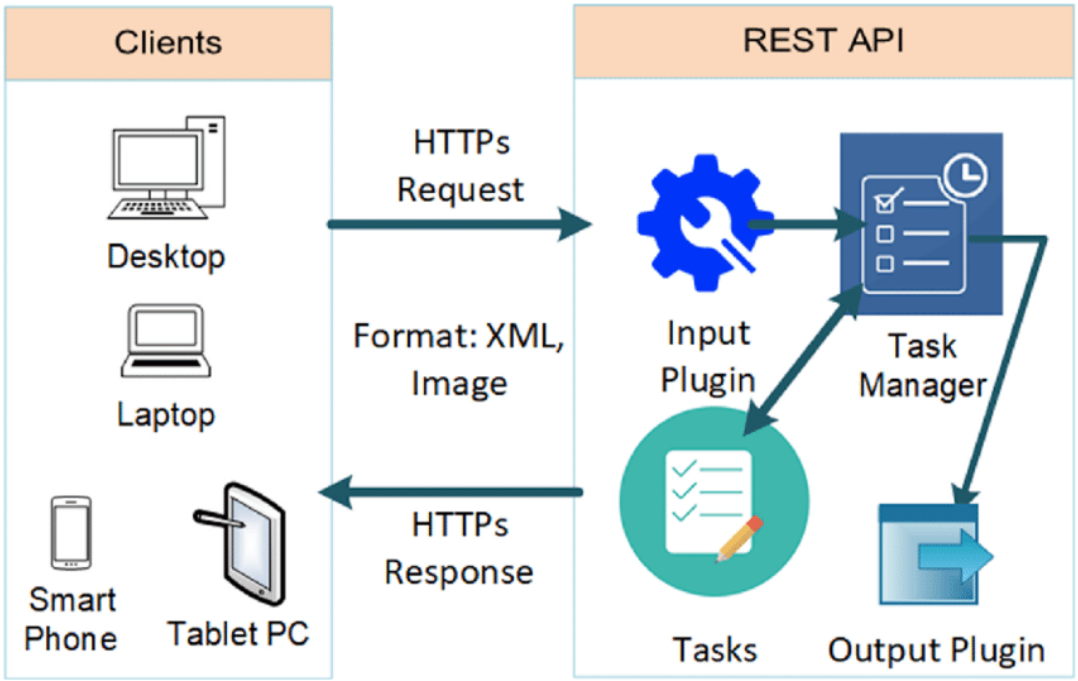

Figure 1 - Application Rest API communication

In terms of technologies, it was a hard decision to be made. And for this application several options were available for the client-side also for the server-side.

For choosing a backend technology more factors were taking into count and more frameworks. The First option for this application as a server-side technology was for sure .NET because it is fast, can be deployed on any

platform and I am most familiar with it, but it has the disadvantage that you cannot create AI components that easy. The Second option was Java with Spring framework, because it is open to OOP writeable code, it is a fast-learning framework and it is very used worldwide, but as well as .NET it does not provide an easy AI solution. Another option was Python because code is very easy to write, it has support for AI, APIs are very easy to write, but as a weakness, it does not provide a strong typing convention.

In terms of client-side solutions the following options were considered: Angular because it is a component-based solution, it has some support from ASP.NET Boilerplate and it is the most used web framework with .NET, but it will generate a lot of files in the end; React.Js because of their growth over time and because it uses components and it has some awesome UI kits available, but the disadvantage is that development starts from scratch. Also, I considered using .NET Razor pages because it does not need two projects, it can bind DTOs directly, but it means that classic styling with CSS and JavaScript should be used.

After research, MvManagement (MvM) is using the latest version of technologies available for now. This application is designed on top of .Net Core 6 which was released on November 8, 2021, according to Microsoft official website [2], for the backend of the application. In terms of client-side technology MvM is using the last version of the React.Js at the moment of writing, one of the most known and used java script framework taking into count the statistic done by Stack Overflow [3]. This statistic shows that React.Js is one of the most preferred and most used web frameworks and it will be clear why after next paragraphs.

Server-side technologies

As mentioned, this project is built on one of the most used technologies all over the word. Why .Net? Because .Net is a widely used solution for developing applications, because it is fast, it is maintained by one of the word's giant companies, Microsoft, allows asynchronous programming. It is cross platform allows deploying on any platform not only on Windows; this concept is supported by Microsoft on their website [4]. Last but not least .Net is open source that means that everyone can contribute add their knowledge in order to get the best from this framework. In terms of programming languages .NET uses C# which is a solution that encourage all the OOP related principles as Encapsulation, Composition, Aggregation, Inheritance, Abstraction and so on.

The version of the .NET was an important decision to be done, and this MvManagement application is built on .NET 6 because it is an LTS (Long term support) release, which means that it benefits from Microsoft support for more time. According to the book "C# 10 and .NET 6 – Modern Cross-Platform Development" by Mark J. Price [5] the LTS versions of .NET are supported since a new LTS version is released and Current versions are supported just for a period after a newer version come out.

Driving without rules is impossible and everything will be a disaster, a lot of crashes, standstills and so on. That can be also applied for coding and there are also some rules and principles for writing. One of the most known and relevant rules and best practices for Object-Oriented programming (OOP) are the five SOLID Principles. Each letter of the word SOLID names one principle. First principle is the Single Responsibility principle, this principle states that a class should do only one thing. The next principle described is the Open-Closed principle, it states that classes from OOP should be open for extension and closed for modification. This means that code inside classes can not be modified but the name class can be extended, more functionalities can be added. The third principle, corresponding to letter L, the Liskov Substitution Principle which states that child classes should be substitutable for the parent classes. Next principle, the Interface Segregation Principle is about separating interfaces. The reason why this principle appeared is because a class should not be forced to implement

methods that are not required. The last principle, corresponding to letter D, the Dependency Inversion Principle states that classes should depend on Interfaces and abstraction not on concrete classes.

MvManagement is built on top of ASP.NET Boilerplate framework [6] created by Volosoft and maintained by the entire community because it is an open-source framework, moreover, it also benefits from the support of the .NET Foundation. It is a Framework design on top of SOLID principles that is a very good solution for developing web application with actual practices and tools. What does it provide? ASP.NET Boilerplate (ABP) from my use and according to their documentation it offers a layered architecture based on DDD (Domain Driven Design), modularity, the possibility to build your own modules on top of their modules, multitenancy, a way of storing data about different entities on the same server named by Gartner Glossary [7] and also by the ABP official documentation. In additions, to this awesome programming features, ASP.NET Boilerplate provides a very useful documentation and a prompt, helpful GitHub community. Some common structures provided by ABP are Dependency Injection, session managing, caching, logging and setting management.

Dependency Injection is a software design pattern which aims to separate the declaration of the dependency from the implementation itself. This way by using dependency injection programs manage to be loosely coupled and to follow the dependency inversion and single responsibility principles. In .Net the design of dependency injection is mainly done by creating and interface which contains the structure of the methods, name of the method, returned type, parameters and their type. Interface is implemented by a class or more classes. Further other classes can use the interface methods without knowing how methods are implemented because it is ensured by the interface that the result will be the one expected. Dependency injection is managed inside MvM by ASP.NET Boilerplate framework. It uses the (Castle Windsor), open source, framework for this and some strict naming conventions. By only using Castel Windsor the dependencies should be mapped manually by declaring a Container and then registering each dependency something like:

```
Component.For<IVehicleAppService >().ImplementedBy<VehicleAppService>().LifestyleTransient()
```

With the help of the ABP this kind of dependencies are automatically registered. The naming convention is that if there is any class that implements an interface called IVehicleAppService and its name contains VehicleAppService postfix it will be automatically registered as implementing IVehicleAppService. Classes can have other names without following the naming convention and can be registered in dependency injection container, but it must be done manually.

Session is another very important feature provided by ASP.NET Boilerplate. Basically, it provides an interface IAbpSession which can be injected and can be used to obtain information about current user and tenant without using ASP.NET's Session according to their documentation [6]. AbpSession defines a few key properties like, UserId which represents the id of the current or it can also be null if there is no current user. TenantId another key property and one of the most used describes the id of the current tenant or null if current user is not assigned to a tenant.

Relational Database

In terms of databased MvM remains stuck to Microsoft technologies and it uses Microsoft SQL Database. MS SQL represents a relational database. A relational database is a collection of data items with predefined relation between them. These items are organized in a set of tables with columns and rows. Each table has a strongly defined structure which describe specific kind of object. Each column states a specific attribute and has a specific type. Each object stored in the table represent and row in that table. Each entry in a relational

database table can be unique identified by a property called primary key. The relation between two rows from deferent tables is described by a foreign key.

SQL states for Structured Query Language. SQL is the mediator used to communicate with a database management system. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform different action on database, such as updating an entry, creating a new entry, querying a specific table.

In order to get an efficient way of creating, mapping and using database models MSSQL is combined with Entity Framework 6 [8]. According to Microsoft, Entity framework is an ORM (Object Resource Mapping) tool that allows CRUD operations without having to write SQL queries. Entity Framework provides two important ways of creating mapping, code-first and database-first. Code-first approach as the name says states that the first done is the code, this means that entities are designed in code using some tools provided by Entity Framework than migrated to database by using EF-cli (CLI = Command Line Interface). After entities are designed migration is added using "Add-Migration Migration-Name", then command "Update-database" in order to apply the migration to the database. The second approach, Database-first, means that firstly the database is designed using specific tools or directly from code and then the mappings are done in code. For the MvManagement the method used is code-first because this way some code can be written, tested, the migration added and then the database updated.

Client-side technologies

For rendering the user interface MvManagement is using React.Js library. This library builds Single-Page application. According to the article "Single-page application vs. multiple-page application" written by Neoteric on medium.com [9] single-page application are web solutions that do not need re-rendering a page in order to display new content. And the biggest advantage of using single-page applications is that all the resources (HTML, CSS, Script) are loading once in the app life. Also, this is a component-based JavaScript library, that means it builds component with their unique management context and this results in complex UI.

An advantage of this library is that it is open-source, and you can easily find support. Facebook from 2021 named Meta, decided to use React as an open to development project from 2013 since now, according to their GitHub release history [10]. Now is one of the most used and preferred web framework by developers taking into count the study done by Stack Overflow [3].

In addition to React.Js for this application uses an UI kit named Chakra UI [11]. This UI library was relatively recent released, in 2020, and is very popular among developers because is easy to use and is lined up with react actual standards. Chakra is offering a way of writing fewer and cleaner code, but also let developers extend component in order to get the best for them.

Components represents the most useful part of using an UI kit and Chakra is providing a lot of them. According to their website at the moment of writing they are offering components from 11 different categories. A few of them had an important role in choosing Chakra UI, layout components that are designed to keep your website appearing on all size of the screen, no meter if people are using it from phone or from a desktop. Another category of components that have a very important role to the aspect of the website is Form components category. Forms components provided by this UI kit are appreciated because they are offering nice looking buttons which can be easily can be customizable. Moreover, it provides already design inputs on top of which developer can add its signature in order to make it looking good. There is more other small component have a great impact to your website as Breadcrumbs, Icons, Alerts, Tooltips and so on.

Installing UI kits can sometimes be very annoying, and you must read a lot of rigid documentations. For Chakra UI this is not the case. Installing it is as easy as wrapping the application tag in a “ChakraProvide” tag. This feature was very appreciated by the developer community.

Optical Character Recognition (OCR)

Optical Character Recognition or known as OCR represents a technology used for recognizing and extracting text within a digital source as images and converting information into a machine-readable form in order to use it for data processing. OCR principle seem to be straightforward, but it's implementation may not be that simple because of the variety of fonts and letter spacing formatting. Even then the most difficult job for OCR systems remains recognizing and extracting handwritten text because there are very few writing styles that match, and most offend it will be different.

The process of OCR involves three main objectives that are reached using a series of steps. These objectives are pre-processing of the image, character recognition and post-processing the output. As expected, the first step of OCR is scanning the document. By scanning we can ensure that the document is well aligned and fit a size pattern. This method encourages the correctness and efficiency of text extraction. Preparing the image, this step is focused on removing imperfections from image. This step aims to create a focus on characters and to sharpen them in order to make text clear. The next step is designed to make the document as simple as possible, and it can be called Binarization. Binarization process states in redefining the image document so that it is bi-component build, containing only black and white colors. Black and dark areas are considered to be text zone and the same time the white and light areas are considered background and are ignored, that technique encourage an optimal recognition of the characters. Another phase and the one for which all pre-processing of the image steps were needed is Recognizing characters phase. This phase process the black areas in order to recognize letters or digits. Mainly, OCR focuses on one character or on a block of text. Pattern recognition and feature detection represents the two algorithms mainly used for recognizing characters. Pattern recognition involves training the OCR software in a way that it can learn the font and format. The software is then used for comparing and recognizing characters in the scanned document. Through Feature detection algorithm the OCR software recognize letters and digits by their features in the scanned document. Features in this context can include number of angled lines, curves or crossed lines. A very common use of OCR is vehicle registration plate recognition and now it is implemented in traffic monitoring camaras. In Figure 2 - OCR Example from PaddleOCR is showed an example found on GitHub at PaddleOCR.

Figure 2 - OCR Example from PaddleOCR

The very specific part of recognizing characters is called OCR Engine. OCR is not a very new technology so there are more OCR Engines available and each of those have advantages and disadvantages. For using some OCR Engines paying a tax is required and taking into count he accuracy of the engine the cost can be higher. Most used solution is Google's Tesseract which can be the best option by the fact that it is free, because it is an open-source OCR Engine. The result of this engine maybe is not as accurate as the ones from ABBY FineReader which is a paid option.

Application

Client-side structure

Frontend key features

.Net project structure

AI Used principles

Application features

Database structure

Application tutorial

References

[1] Movcar SRL, "MOVCAR Website," 2021. [Online]. Available: https://movcar.app/.

[2] ".NET and .NET Core Support Policy," Microsoft, 2021. [Online]. Available: https://dotnet.microsoft.com/en-us/platform/support/policy/dotnet-core.

[3] "Stack Overflow insights," Stack Overflow, 2021. [Online]. Available: https://insights.stackoverflow.com/survey/2021#most-popular-technologies-webframe.

[4] "Microsoft - What is .NET?," Microsoft, [Online]. Available: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet.

[5] M. J. Price, "C# 10 and .NET 6 – Modern Cross-Platform Development," in *C# 10 and .NET 6 – Modern Cross-Platform Development*, 2021.

[6] "ASP.NET Boilerplate," Volo Soft, [Online]. Available: https://aspnetboilerplate.com/.

[7] "Gartner Glossary - Multitenancy," [Online]. Available: https://www.gartner.com/en/information-technology/glossary/multitenancy.

[8] "Entity Framework 6," [Online]. Available: https://entityframework.net/.

[9] Neoteric, "Mediu.com - Single-page application vs. multiple-page application," [Online]. Available: https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58.

[10] facebook, "React repository," Meta, [Online]. Available: https://github.com/facebook/react/releases.

[11] "Chakra UI," [Online]. Available: https://chakra-ui.com/.