MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

STUDENT DORM ACCOMMODATION APPLICATION

DIPLOMA THESIS

Graduate: **Andrei Zoltan BONYHAI**

Supervisor: **As. Eng. Iulia STEFAN**

2017

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DEAN
Prof.dr.ing. Liviu MICLEA

HEAD OF DEPARTMENT
Prof.dr.ing. Honoriu VĂLEAN

Graduate: **Andrei Zoltan BONYHAI**

Student dorm accommodation application

1. **Project proposal:** *Development of an application that speeds up the dorm accommodation process for the students of the Technical University of Cluj-Napoca and can replace the current used application.*

2. **Project contents:** *Introduction, Bibliographic research, Analysis and Theoretical Foundation, Detailed Design and Implementation, Testing and Validation, Installation Guide, Conclusions, Bibliography.*

3. **Place of documentation:** *Technical University of Cluj-Napoca, Automation Department*

4. **Consultants:** *As. Eng. Iulia Stefan*

5. **Data of issue for the proposal:** *October 15th 2016*

6. **Handover date:** *July 11th 2017*

Graduate ___________________________________

Supervisor _________________________________

MINISTERUL EDUCAȚIEI NAȚIONALE

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

Declaraţie pe proprie răspundere privind autenticitatea proiectului de diplomă

Subsemnatul(a) **Andrei Zoltan BONYHAI**, legitimat(ă) cu CI/BI seria MM nr. 794944, CNP 1940511245022, autorul lucrării:

Student Dorm Accommodation Application

elaborată în vederea susţinerii examenului de finalizare a studiilor de licență

la **Facultatea de Automatică și Calculatoare**,

specializarea **Automatică şi Informatică Aplicată (în limba engleză),**

din cadrul Universității Tehnice din Cluj-Napoca,

sesiunea Iulie 2017 a anului universitar 2016-2017,

declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancțiunile administrative, respectiv, *anularea examenului de licenţă*.

Data	Andrei Zoltan BONYHAI
________	______________________
	(semnătura)

Contents

1 Introduction

1.1 Project context

Although we live in a high tech era, the bureaucracy, paperwork, and queues did not reduce. Even if the technology we have at hand is more advanced than ever, the queues in some institutions, in example the city hall, the police office, hospitals, and insurance companies did not shorten at all, especially in rush hours.

The queues have slowly became a part of our daily lives: in hypermarkets, banks, ATM machines and so on. The bureaucracy has a key role in the growth of the queues in different institutions. The number of documents required for a wide range of approvals from institutions are the first thing a person runs into when they need some signed documents by a public institution. This happens especially while signing up for your driver's license, where you need numerous notices: psychological, ophthalmological and so on but as well when enrolling into a university where you need your high school diploma, birth certificate, ID card, and so on. The problem that still persists is that there are periods of time when a really high number of people are requesting a really high number of documents from the same institutions.

There are some methods through which the shortening of the time spent in a queue was attempted. Some of these methods are the following: paying with your credit card in hypermarkets, without the cashier having to calculate the change you should receive. One other method for this are the queue spot machines used in most of the Romanian public institutions today. When you arrive at the queue, you punch a button on the machine and a small card with your queue number is released. Doing this you can get a hint of how much time you have to wait before your turn comes. Taking in consideration that you wait for a while to check how long the persons currently in line wait, in example 30 minutes per person, you could calculate the number of hours you would have to wait and return back to the institution when your time has come. Although this seems like a good idea on paper, that period could vary a lot. In example, you have ten people ahead of you, and each should take up about 30 minutes to finish, probably some of them will encounter some other difficulties and take more than 30 minutes, so when you return to the queue just before your turn you will find out that you have to wait an extra hour in the best case.

Every year, the students who need to live in the university's dorms are put through a constant struggle of waiting in queues for hours, endless paperwork, and frustration. The accommodation process should be relatively simple, but due to the current used application, which is old and rudimentary, it can become really stressful as well for the students and the dorm committee and administrator as well.

Based on author experiences related to the dorm accommodation process as a committee member for the past two years, several issues which made the entire process end up in a bottle-neck were identified. In both these years, the current application crashed, resulting in lost data and having to repeat the process all over again. Aside of that, the only ones who could add the students' data were the dorm administrators.

The student has to arrive at the dorm he was placed in with his file of papers required for the process. Then the student receives an order number. When his turn finally comes the accommodation process begins. In the best case scenario, the dorm administrator assigns him a room. In the worst case scenario, the student is not featured in the database and the administrator or a committee member has to add all the student's data in the application and only after that the student is assigned to a room. After being assigned to a room, his file is processed and his documents are stored. The next step is to have a custom contract for the student printed out, and then have a committee member fill out a badge so that the student can prove he lives in that particular room. After all these steps are over, the student then proceeds to make the payment for the first month. The entire process could take up to 4 or 5 hours for a student (including his waiting time).

In order to reduce the time spend waiting in queues in order to get accommodated a new accommodation web application was proposed in order to shorten the accommodation process for each student.

The web application's role is to reduce the number of steps required going through before a student receives the key to his new room in the dorm he was assigned to and it gives the student the opportunity to enter the relevant information on his own before even joining the queue. So, as mentioned above, this way the time spent by a student in order to get accommodated should be more constant now since the possibility of not having his information in the database and having the administrator adding it by hand is diminished.

The application's use should shorten the entire process' duration by at least 60% for each user, therefore reducing the stressful waiting in lines, and as well helping out the dorm administrators and committees since they won't be required to work for 12 – 13 hours a day during those 4 or 5 days of the accommodations.

1.2 Objectives

The proposed project consists in a web application and a QR code that should be used by the students waiting in the queue in order to register their information in the database before their turn comes.

The application's purpose is to make the accommodation process of each student easier by using a user-friendly interface that can be easily used by the accommodation committee.

One of the objectives is the creation of a REST service that can be used by a wide range of devices, therefore increasing the number of users without keeping in mind the model of the used device that would be used in the process. For the data transmission the JSON format will be used. JSON is a platform independent format and so it facilitates the use of the application on a wide range of devices without depending on the device model or the operation system on the respective device. Using this REST service the number of additional implementations for more platforms will be reduced to zero.

Another objective is the development of a web application that ensures a fast paced processing of a student. Usually the accommodation flow in any dorm is known. This process takes longer when all the information regarding a student have to be added by hand and so creating a bottle-neck in the process, because the student next in line should wait more time until the administrator adds the data. And taking in consideration that most of the dorm administrators' age is over 50, many of them are not so familiar with the use of a computer, therefore they type in the data way slower than usual.

Another important aspect is having the student upload his documents in digital version while registering his information. In this case, when the respective students' turn in the queue is up no additional paperwork processing would be needed. The administrator would just assign him to a room, generate the contract and the student should only pay the first month rent.

1.3 Specifications

One of the application's objectives is having a simple, crisp, and user-friendly interface. This is an important factor because the application should offer the possibility of shortening the duration of the process, and in order to do this, the administrator and the accommodation committee should get used to the application and the steps required going through in order to increase the speed of the accommodation process.

A register account and authentication page is required so that each dorm administrator and committee member can have a username and a custom password. The password should be encrypted using an encryption algorithm so that in the case of someone unauthorized gaining access to the database intending to damage the system by any means the current existing users' passwords should be protected.

One of the key objectives of the application is creating a page where the dorm overview is available, in order for the administrator to see the status of each room: available places in the room and other students from that room.

A page containing a table with all the students in the database and the option to assign them to a room in the dorm as well as a search option in order to get rid of the scrolling until a respective student is found in the table should be implemented.

Another objective of the application is to generate a QR code that can be scanned by students so they update their information in the database on their own. Scanning this code would redirect the student to a web page of the application containing a registration form and a file uploader for the documents. This page, besides the other pages of the application, will not require to be authenticated with and existent user in order to be accessed

Another aspect of the QR code linked page is having certain restrictions on the page with the URL encoded in the QR code is demanded because students should not have access to any other private information inside the application such as other students' data, or the possibility to select a room for themselves before the administrator decides.

2 Bibliographic research

2.1 QR Codes

The QR code(abbreviation from Quick Response code) is a matrix bar code(or a bidimensional bar code) that was initially used in the Japan railway industry. This system was developed by Denso Wave in 1994 with in order to keep track during the fabrication process of the vehicles. The QR code was conceived to allow fast scanning.

The QR code is conceived to be scanned by a bidimensional video sensor and then being processed, unlike the standard one dimensional bar code which was conceived to be scanned by a narrow beam of light. The processor identifies the three distinct squares placed in the corners of the QR code image and uses one, or more, small squares in the fourth corner of the code in order to determine the orientation, scanning angle and size. The small dots in the QR code are then converted in binary code and validated by an error correction algorithm.

The amount of information stored on a QR code depends on the data type(binary, alphanumeric, numeric, kanji) as well as on the error correction code.

The error correction in a QR code is achieved using the Reed-Solomon error correction algorithm. A higher error correction level translates into a smaller amount of information encoded on the respective QR code. In the case of large dimension QR codes the information is split in more Reed-Solomon blocks. The dimensions of these blocks is chosen such as the number of errors that can be corrected in one block does not exceed 15 therefore limiting the complexity of the decoding algorithm. The Reed-Solomon blocks will be overlapped so that the errors in one QR symbol will not overwhelm the capacity of that block. Due to the error correction one can insert errors into the QR on purpose to make them more aesthetic for the human eye [1].

2.1.1 Classification

Micro QR is a smaller version of the standard one designed for application where the dimension of the symbol is limited. There are 4 versions of the Micro QR code: from the largest one being able to store up to 35 numerical characters to the smallest one composed of 11x11 modules.

IQR code was developed by Denso Wave as an alternative to the standard QR code which can be used in square or rectangular shape. The rectangular shape comes in handy when the object which the QR is generated for has a cylindrical shape. These codes can store the same quantity using only about 70% of the space. Today there are about 61 variations of square IQR and around 15 versions of the rectangular one.

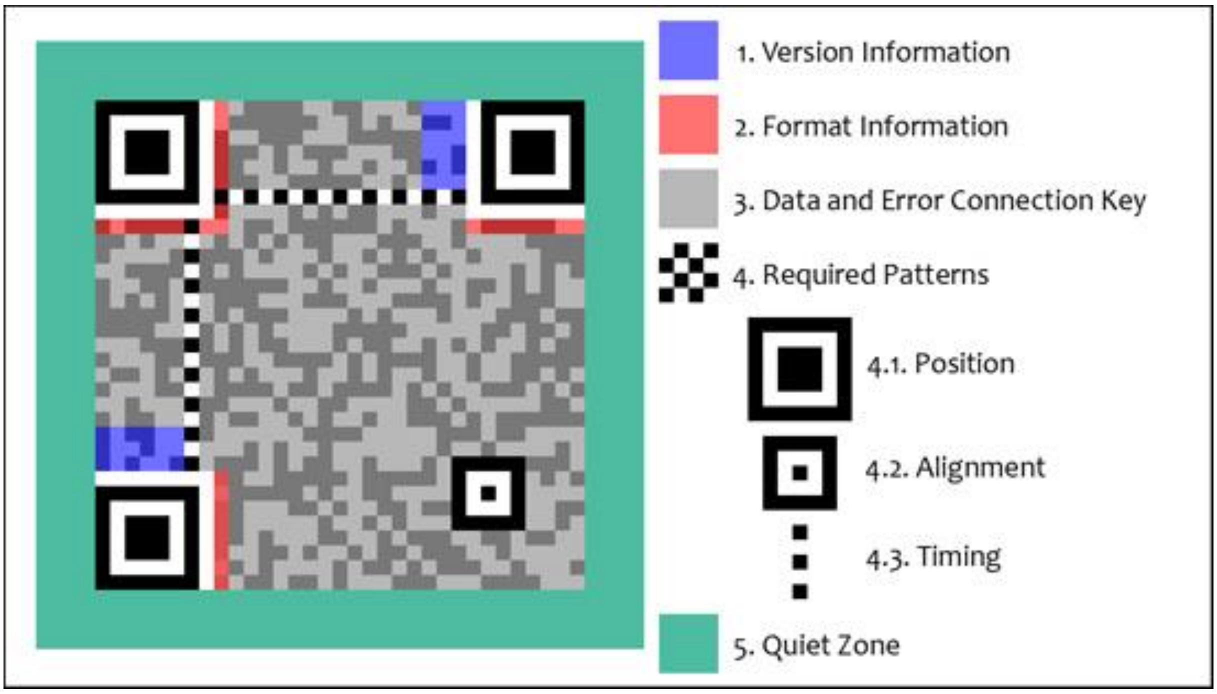


Figure 2.1 QR code structure

2.1.2 Risks

The only case in which standard QR codes can have executable data is the URL data type. These URLs may contain JavaScript code that can exploit vulnerabilities on the reader or the web browser.

If the software is not exploited, having a permissive reader scan a harmful QR code can expose the computer's content and the user's private information. This process is known by the name of "attagging". On a smartphone it can gain complete access to the internet, permission of using the camera, reading and writing local data and globally modifying the system's data.

Scanning a malicious QR code can lead to accessing unsecure websites which exploit various browser vulnerabilities allowing access to the phone's camera, microphone and GPS and then transmitting sensitive data(passwords, files, contacts, transactions) to a server in order for them to be analyzed. It can as well send e-mails, SMS messages, Instant messages or DDOS packages as part of a network of private computers all being infected in order to corrupt the confidentiality settings and leading to identity theft. All these intrusive processes can run in the background while the user thinks he opened a harmless web page[2].

2.1.3 Uses

2.1.3.1 Mobile devices' operation systems

These operation systems support URL redirecting, allowing QR codes to transmit metadata to other apps already installed on the device. A wide range of free or purchasable applications have the option of scanning QR codes and redirecting to specific external URLs.

2.1.3.2 QR code payment

The QR codes can store the information of a bank account or a credit card account and can as well be specially designed to work with a specific payment provider's applications.

The QR code payment was broadly used in the Czech Republic in November 2012 when an open format for exchanging payment information was released and approved by the Czech Banking Association as an official solution for paying using QR codes.

QR codes are often used in the field of cryptographic currencies such as Bitcoins. Cryptographic keys, transaction information and other information are shared between digital wallets this way.

2.1.3.3 Encryption

An android app can encrypt and decrypt QR codes using the DES algorithm. One particular case where encrypted QR codes are used is in the Japanese Immigration system, where they are used for passport visas. Otherwise, encrypted QR codes are not generally used.

2.1.3.4 Funerary use

Ishiniokoe in Japan started selling tombstones containing QR codes in 2008. Those codes would lead to a digital grave site of the deceased.

2.1.3.5 Website authentication

The website generates a QR code and the user can scan that code inside a mobile app. After scanning the code, the user is authenticated on the website as well. This procedure is used by Whatsapp web application.

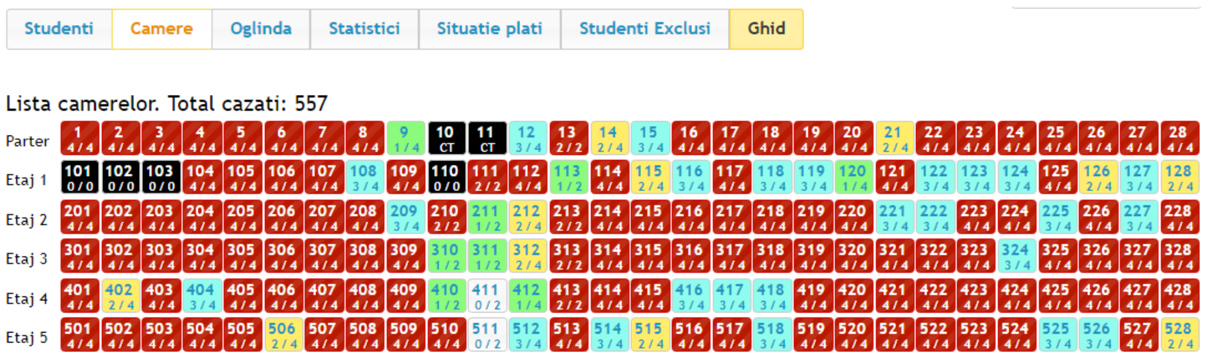
2.1.3.6 URLs

Since the QR code is a faster method of accessing a website than manually entering the URL it quickly became an interest for the publicity industry. Excluding the simple commodity of the fast website access, publicity companies hope to turn the ad into a selling process, by bringing the user to the company's website.

2.2 State of the art

2.2.1.1 Current student dorms accommodation app used by the TUCN.

The current student dorms accommodation app is a web application used by the Technical University of Cluj-Napoca during the dorm accommodation period. It is linked to the databases of the University's department of Informatics.



Lista camerelor. Total cazati: 557

Parter	1 4/4	2 4/4	3 4/4	4 4/4	5 4/4	6 4/4	7 4/4	8 4/4	9 1/4	10 CT	11 CT	12 3/4	13 2/2	14 2/4	15 3/4	16 4/4	17 4/4	18 4/4	19 4/4	20 4/4	21 2/4	22 4/4	23 4/4	24 4/4	25 4/4	26 4/4	27 4/4	28 4/4
Etaj 1	101 0/0	102 0/0	103 0/0	104 4/4	105 4/4	106 4/4	107 4/4	108 3/4	109 4/4	110 0/0	111 2/2	112 4/4	113 1/2	114 4/4	115 2/4	116 3/4	117 4/4	118 3/4	119 3/4	120 1/4	121 4/4	122 3/4	123 3/4	124 3/4	125 4/4	126 2/4	127 3/4	128 2/4
Etaj 2	201 4/4	202 4/4	203 4/4	204 4/4	205 4/4	206 4/4	207 4/4	208 4/4	209 3/4	210 2/2	211 1/2	212 2/4	213 2/2	214 4/4	215 4/4	216 4/4	217 4/4	218 4/4	219 4/4	220 4/4	221 3/4	222 3/4	223 4/4	224 4/4	225 3/4	226 4/4	227 3/4	228 4/4
Etaj 3	301 4/4	302 4/4	303 4/4	304 4/4	305 4/4	306 4/4	307 4/4	308 4/4	309 4/4	310 1/2	311 1/2	312 2/4	313 2/2	314 4/4	315 4/4	316 4/4	317 4/4	318 4/4	319 4/4	320 4/4	321 4/4	322 4/4	323 4/4	324 3/4	325 4/4	326 4/4	327 4/4	328 4/4
Etaj 4	401 4/4	402 2/4	403 4/4	404 3/4	405 4/4	406 4/4	407 4/4	408 4/4	409 4/4	410 1/2	411 0/2	412 1/4	413 2/2	414 4/4	415 4/4	416 3/4	417 3/4	418 3/4	419 4/4	420 4/4	421 4/4	422 4/4	423 4/4	424 4/4	425 4/4	426 4/4	427 4/4	428 4/4
Etaj 5	501 4/4	502 4/4	503 4/4	504 4/4	505 4/4	506 2/4	507 4/4	508 4/4	509 4/4	510 4/4	511 0/2	512 3/4	513 4/4	514 3/4	515 2/4	516 4/4	517 4/4	518 3/4	519 4/4	520 4/4	521 4/4	522 4/4	523 4/4	524 4/4	525 3/4	526 3/4	527 4/4	528 2/4

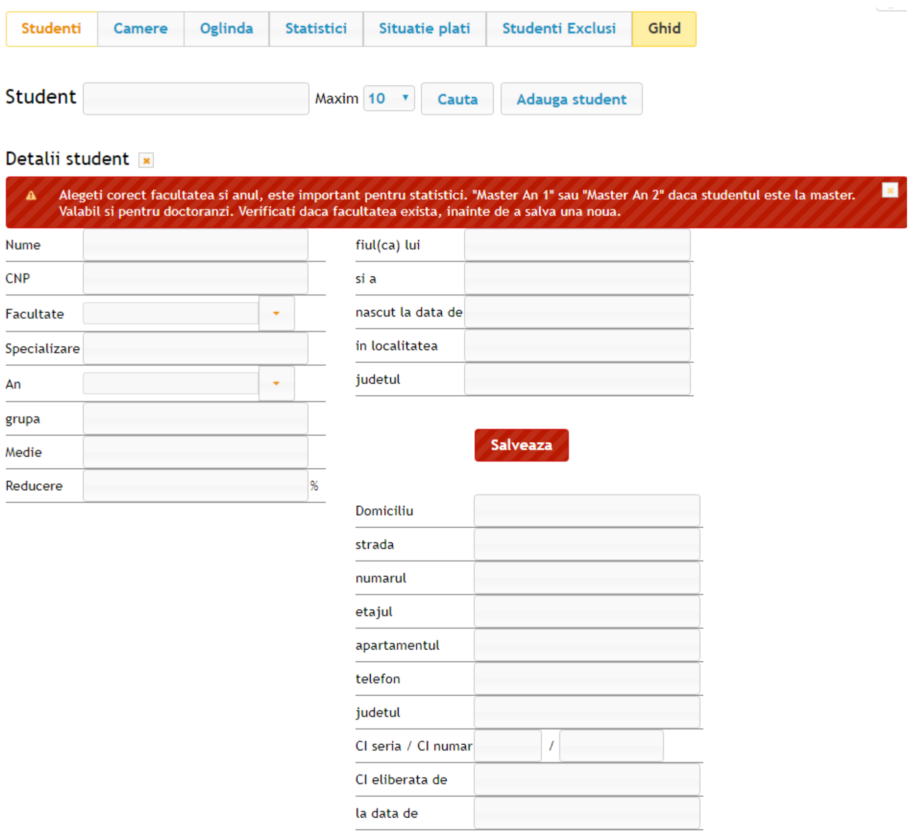
Figure 2.2.1.1 Current application dorm overview

This application is pretty outdated, having a pretty old user interface which is not only unappealing but also pretty hard to get used to. But one of the biggest issues of this application is that the user accounts passwords are not encrypted at all. In the User table of the database you can see the username and the password string without encryption. In case someone manages to access the database he can cause a lot of harm to the system without too much trouble because he can simply copy and paste the password from the table.

Every year the current application crashes resulting in losing all the data from the database and the accommodation has to be restarted by going through each file for each student that has already been accommodated and re-adding their data in the database then assign the rooms previously given to them. This usually leads in 9 or 10 dead hours in which no student will get accommodated and the queue will only get longer and longer plus that the process can be delayed which is really unpleasant.

The Pharmacy and Medicine University in Cluj-Napoca is using an online application for the dorm accommodation process. In their application the student can even chose which room he wants to be accommodated in.

Other technical universities across the country such as the one in Iasi and Timisoara are using a similar accommodation application like the Technical University of Cluj-Napoca.



Alegeti corect facultatea si anul, este important pentru statistici. "Master An 1" sau "Master An 2" daca studentul este la master. Valabil si pentru doctoranzi. Verificati daca facultatea exista, inainte de a salva una noua.

Figure 2.1 Current student registration form

3 Analysis and Theoretical Foundation

3.1 ASP .NET MVC

3.1.1 The MVC pattern

The MVC pattern was used for developing this application in order to decouple the business logic from the data models and the views.

Model-View-Controller (MVC) is an architectural pattern which influenced the computer science domain over the last decades. Originally named Thing-Model-View-Editor in 1979, it was later on simplified to Model-View-Controller. It's an elegant and powerful method of separating the logic in an application which works really well with web applications. It is explicit concern separation increases the complexity level of an application's design by a small amount but the extraordinary benefits justify the extra effort. Since it was introduced, the MVC pattern has been used in more and more frameworks. MVC can be found as well as Java and C++, on MAC OS and windows and it can be found as well included in a broad variety of frameworks.

3.1.2 ASP .NET MVC

ASP .NET is a web platform providing all the necessary services in order to build an enterprise server based web app. ASP .NET is developed based on the .NET framework so that all the framework's facilities are available to the ASP .NET applications

Web applications developed with ASP .NET are ran by the .NET framework and not by the OS. This makes the apps type-safe and has the advantage of having automated memory for garbage collection. In addition, the .NET framework provides us with a structured error management and supports multithreading. Information about classes, members, and in general, all our code is stored as metadata in kits generated during the compilation.

To launch ASP .NET web applications one can use one of the various existing techniques, in example Web Deploy, or the most basic method, copying of the files onto the server. After all the applications are on the server, the Microsoft web server (Internet Information Services or IIS) must be configured. The previously mentioned server will host all our ASP .NET applications and will make them available to the clients.

One notable fact is that ASP .NET is completely object oriented. The written code will have access to all the objects in the .NET framework and it also can implement all the benefits of an object oriented development environment (inheritance, polymorphism and encapsulation). [3]

ASP .NET MVC is a web application development framework which is using the Model-View-Controller pattern. It is developed on top of the ASP .NET framework. The main purpose of this pattern is to isolate the business logic from the interface for a better maintenance, improved testing and simplifying the structure of the application.

The Model-View-Controller (MVC) pattern is an architectural design principle which separates the application's components into three levels. This separation offers more control over the individual parts of the application, allowing a faster and better development, improvement and testing of the specified parts.

The MVC pattern in ASP .NET would consist of the following:

- The Model: The classes representing the domain of work. Often these objects encapsulate data stored inside a database as well as the code for the data manipulation and imposes the business logic. In ASP .NET MVC this would most likely be the Data Access Layer
- The View: A template for dynamic generation of the HTML. One could say that the view is responsible for transforming one or more models into a visual representation
- The Controller: A special class managing the relationship between the view and the model. It reacts on the data entered by the user, communicates with the model and decides which view should be displayed (in the case the view exists). In ASP .NET MVC this class is conventionally named with the "Controller" suffix.

A representative image for the MVC architecture can be seen in figure 3.1 below

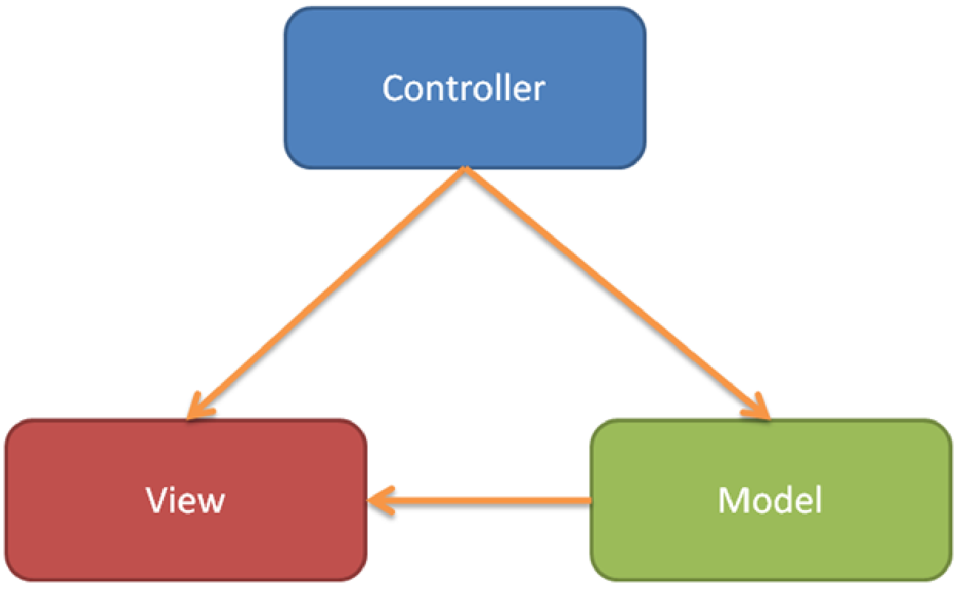


Figure 3.1 MVC Architecture

ASP .NET has certain abilities that make it the best option if you have to choose between one and many of the following:

- High level of control over the generated HTML. The ASP .NET MVC view will be generate exactly how the code was written.
- Easier unit testing
- Separating concerns. Having all the aspects of the system clearly separated one from another. Because of the pattern it implements, a MVC application is split in different decoupled parts (model, views, controllers), therefore permitting a much more simplified maintenance process.

One of the most important concepts of the MVC application is that there is no relation between requesting a page and a certain file on the server.

In MVC applications, when a request is made, a component named routing engine attempts finding the correct route for the current request. A route defines the requests using a string as a pattern and decides which controller and which method in the selected controller should process the request. Once the route is identified the routing engine creates a request handler, which will create a Controller type object to process the requests. The controller will then call the specific method for the requests processing. These methods found in the Controller classes are called action methods. When a request is done being processed, the action method will come up with a result to be passed back to the user. Usually, the result is a HTML code which the user will see in the browser.[4]

3.1.2.1 JSON

JSON (JavaScript Object Notation) is a standard for data exchange. There is a broad range of parsers for different languages. JavaScript objects can be represented in JSON and can be sent as a part of an Ajax request. The same way if an action method in a controller returns a JavaScript Object, MVC is capable of converting the object in JSON in order to be returned with the response.

JSON is usually preferred over XML for data exchange between the browser and the server because JSON can represent almost every data structure XML can but with a lesser load.[6]

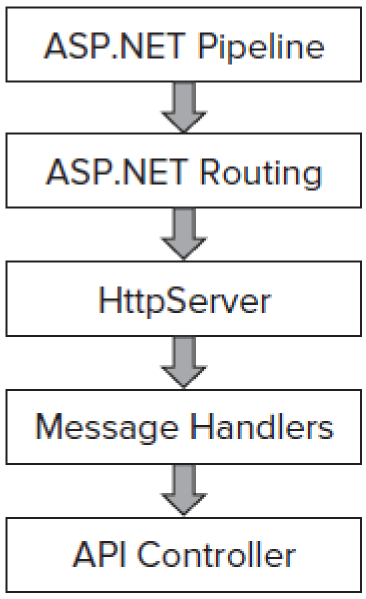


Figure 3.2 Web API Architecture

3.2 ASP .NET Web API

The ASP .NET Web API was used to handle the HTTP requests of the application and for returning the information in the database.

ASP .NET Web API is a framework that facilitates building HTTP services for a broad range of clients, including mobile devices and browsers.

Web API is an ideal platform for the construction of pure HTTP services where the requests and response are generated using the HTTP protocol. The client can call one of the following requests: GET, PUT, POST, and DELETE, and the Web API will return the corresponding response.

3.2.1 Web API Architecture

The routing configuration in Web API is a bit different than the routing in ASP .NET MVC because Web API is using HttpRouteCollection and Route. The Web API team reused the routing logic from MVC, but this one is different due to desiring to keep it independent of the ASP .NET routing. They decided to make it independent so that Web API will not have any dependencies of the ASP .NET classes and so it can be used in the Windows console or a Windows service.

While using the ASP .NET routing the system won't only register the HttpRoute object, but will as well create a Route wrapper object.

The difference between ApiController and the MVC controller consists in the fact that in the WebApiController the requests are automatically sent to the corresponding actions according to the type of the HTTP request. There is still some flexibility left, allowing the overwriting the previous mentioned fact so that WebApi will use the action's name in order to redirect the requests to the corresponding WebApiController method.[4]

3.2.2 Content negotiation in Web API

Content negotiation is the process implying the selection of the data’s best representation for a response when more representations are available.

The framework Web API is based on allows the client to select a specific data type on request because it implements content negotiation.

Web API responds using the JSON format by default, although when making a request to a certain resource you can specify the returned data type so that Web API will know what is requested and choose the requested data type.

3.2.3 Routing in Web API

In order to determine which action should be called, the framework uses a routing table which is similar to the one used in MVC.

The Web API framework receives a HTTP request, and it tries to match the URI with one in the routing table. If no route in the table matches, the client will receive an error 404.

The simple data type parameters are taken directly from the URI, but the more complex parameter types are taken from the request body, the body being deserialized based on the sent content.

3.3 Entity Framework

Entity Framework was used in the application in order to have the data structured in objects and properties and avoid writing long SQL queries.

Entity Framework is a set of technologies from ADO .NET that support the development of data oriented software applications.

Entity Framework allow the developers to work with the data structured as objects and proprieties specific to the domain, such as clients and their addresses without bothering with database tables and columns that store the data. With Entity Framework the developers can use a higher level of abstractization while working with the data. They can create and maintain data oriented applications with less code required than in traditional applications. Since Entity Framework is a component of the .Net framework, the applications using Entity Framework can run only on a computer with the .NET framework (minimum version 3.5 SP1) installed.

3.3.1 Architecture

The ADO .NET architecture contains the following:

- Data source provider using the database connection interfaces when coding on the conceptual schema
- Map provider, a specific provider for each database translating the Entity SQL command tree in a native SQL query for the database
- The data model sends its specification to the EDM Parser and then the view mapper maps them to the relational model. This allows the development on the conceptual model. A visualization of the data corresponding to this model is created using the relational schema, and then an entity is created by aggregating information from two or more tables.
- Queries and updates: filters and updates the requests after the query has been processed in order to convert them into canonical command trees, which are then converted into native database queries by the mapping provider.
- Metadata services: handling the metadata related to the entities, relationships and mappings
- Transactions: in order to integrate with the transactional capacities of the used database
- Conceptual layer API: exposes the programming model in order to code on the conceptual schema. It uses the ADO .NET connection type object pattern in order to refer to the mapping provider using Command type objects to send the query and return EntityResultSets or EntitySets containing the result.
- Embedded databases: ADO .NET Entity Framework includes an easy to embedded database for caching and querying relational data on the client side
- In order to simplify the mapping from the conceptual to the relational schema, a variety of design tools are embedded in the ADO .NET Entity Framework. One of these tools is the Mapping Designer which can specify the correspondence between the properties of the entity and the database table.
- The EDM is used as a programming construct by a special layer (programming layer) in order for it to be used by the programming languages.
- High level services such as reporting services working with entities rather than relational data

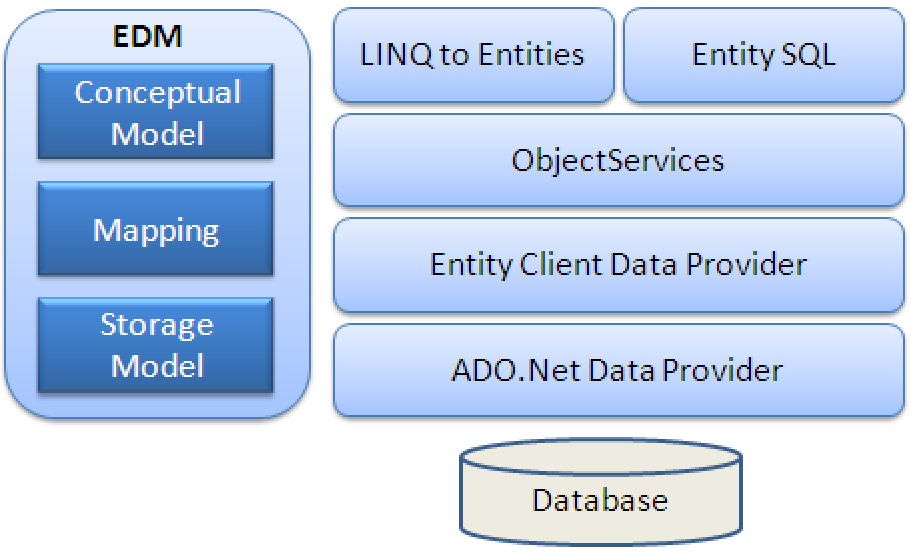


Figure 3.3 Entity Framework Architecture

3.3.2 Entity Data Model

The Entity Data Model or EDM is a technique used for specifying the conceptual data model. It is a more general version of the Entity Relationship Model. It mainly describes the entities themselves and their associations. The Schema Definition Language, or SDL, is used in defining the Entity Data Model.

Visual studio contains Entity Designer, in order to visually create the EDM and specifying the mapping. Using the Entity Designer results in XML files with the ".edmx" extension specifying the schema and the mapping

3.3.3 Mapping

EDM Wizard in Visual Studio initially maps the database and the conceptual schema in a one to one relationship. The relational schema is made of elements that are in fact keys linking related tables.

The entity types are an aggregation of multiple typed fields. Each one maps a column of the database and can contain information for multiple tables. The entity types can be related to each other and independent of the relationship in the physical schema. The related entities are exposed similarly using a field describing the relation they are participating in. Instead of retrieving the value from a column, it returns the entity or a collection of entities.

An EDM structured as an XML is used to illustrate the logical schema and the logical schema's mapping with the physical one. This XML is used for the mapping and it allows the project to use the entities.

The Entity Framework handles the table joins required for an entity to have information referenced across multiple tables. When a certain entity is updated, a backtracking process is started in order to perform the SQL queries and to update the values in the tables where the information came from. For the theoretical operations and queries the Entity Framework uses a derivative scripting language based on SQL, also known as eSQL that can be translated back into native SQL if required.

3.3.4 Entities

Entities are instances of Entity Types. They represent individual object instances (such as client, and commands). An Entity's identity is defined by the entity type it is an instance of. The properties of an entity type in ADO .NET Entity Framework are fully typed and fully compatible with the type system used in a DBMS system as well as the Common Type System of the .Net Framework. A property can be SimpleType(corresponding to primitive data types such as Integer and Characters) and ComplexType which is an aggregate of multiple SimpleType propreties

3.3.5 Relationships

Association or Containment can be used to relate any entity types between each other. An action or an operation can be associated with any relationship type. This allow the action to be performed on the parent entity in case of another action being performed on the child entity or the other way around.

3.4 Simple Injector

The Simple Injector library helps with the dependency injection in the project. It prioritizes the loose coupling of the elements.

Simple injector is a Dependency Injection (DI) library very small, compact and simple to use that can be easily integrated with technologies such as Web API, MVC, WCF, ASP .NET, ASP .NET Core and many others. It implements the dependency injection pattern using loosely coupled components.

The basic features of integrating the Simple Injector library with the ASP .NET Web API are the SimpleInjectorWebApiDependencyResolver class and the RegisterWebApiControllers extension method.

The Dependency Injection is a design pattern that demonstrates how to loosely couple classes. Even if you use the dependency injection pattern to loosely couple classes you might still wonder why it is so useful. One of the best utilities for this pattern is the ability to quickly create and perform Unit Tests, since every component will be defined by an interface which it inherits. Using these particular interfaces one can then create new classes identical to the ones used in the project in order to perform unit tests and validation/exception management. [11]

3.5 ECMAScript 2015

It is a standardized specification of a language developed by Brendan Eich. The programming language went through a few naming stages. First it was Mocha, then LiveScript and finally JavaScript, the name it is known by today.

ECMAScript is based on a few technologies. The most well known of these technologies are JScript developed by Microsoft and JavaScript developed by Brendan Eich at Netscape.

The Standard was adopted under the fast-track procedure and it was approved as an international standard (ISO/IEC 16262) in 1998.

The ECMAScript Language Specification development was started in November 1996 and the first edition was adopted by the Ecma General Assembly in June 1997.

The sixth edition of ECMAScript is known as ECMAScript 2015. It added new syntax (classes, modules, etc.) for developing complex applications. While ECMAScript 5 defined the syntax in a strict mode, the new ECMAScript defines it semantically.

Some other new and important features are the promises, typed arrays, arrow functions iterators and for/of loops. Of course these are only a few features, the complete list being extensive.

Browser support for ECMAScript 2015 is still incomplete but the code can easily be transpiled to ES5 code which has a more consistent support across browsers.

ECMAScript is object-based. An object is a collection of zero or more properties each having attributes that determine the use of the property. ECMAScript defines a collection of built-in objects that define ECMAScript entities. Large ECMAScript programs are supported by modules allowing the program to be divided into multiple sequences of statements and declarations. Each module explicitly identifies the declarations it uses that have to be provided by other modules and which are available to other modules as well.

ECMAScript syntax resembles the Java syntax. It is a relaxed syntax in order to serve as an easy-to-use scripting language. For example, a variable does not require a type defined for it.

3.6 TypeScript

For the project TypeScript was used in order to have a more structured JavaScript code.

TypeScript is a superset of JavaScript. In essence TypeScript is still JavaScript but with slight improvements to the code writing. It provides several features that allow you to write more robust JavaScript code:

- Static typing - Every property, function and field can be decorated with type declaration. JavaScript does not care about type declaration, and so this feature of TypeScript is helping developers to have a consistent code using the same types for variables and functions all across the project.
- It is optional – TypeScript is completely optional unlike the majority of other programming paradigms. There is no need to use all the features of TypeScript if you don’t need them. You can even write standard JavaScript code inside the TypeScript files and the output will be the same.

- Free and open source – TypeScript is free even for commercial use, it is available on all operation systems and not limited only to the Microsoft stack. TypeScript can be installed via the NPM package manager, NuGet package manager or even downloaded from GitHub
- Write modern JavaScript – It allows the developers to write code according to the emerging standards and still maintain the backwards compatibility. Technically TypeScript is a transpiler and not a compiler with a lot of useful transformations.

In conclusion, TypeScript is a tool that allows developers to write a code that is more robust, scalable and maintainable. It also puts behind us the days when the maintenance of large JavaScript applications ended up in thousands of lines of spaghetti code that can only be maintained by a single developer understanding all the parts.[6]

3.7 AngularJS

Angular is the main framework used for the development of the front-end part of the application.

AngularJs is an MV JavaScript framework. Since you may already be familiar with the MVC architecture which is model/view/controller it must be pointed out that in the Angular's case as well as many other frameworks, the controller part is not pretty well defined. The controller is not the middle man between the view and the model like in the standard backend MVC architectures but it is rather a scoped object containing functions and logic helping the application interact together.

One has to keep in mind that Angular is a front-end framework for running single page applications. This makes it special because it can load new data and interact with it without having to reload the page or having the server figure out how to display the data.

The use of data binding and dependency injection helps eliminating a part of the code that otherwise would have to be written. Since all of this happens within the browser this makes Angular to work great with any server side technology.

The mismatch between static documents and dynamic application is solved with either a library or a framework. A library is a collection of functions used while writing web apps. The code calls into the library when it is needed (in example jQuery). A framework is basically a web application template where the custom code fills in the details. Unlike the libraries, the framework is in charge and calls the custom code when needed.

Angular attempts to minimize the impedance mismatch between HTML and what the application really needs by creating new HTML constructs. Angular teaches the browser new syntax through directives.[7]

3.7.1 Main advantages of AngularJS

Angular presents a higher level of abstraction and so it simplifies the application development but the cost of this abstraction is the flexibility. During the process, the developers created AngularJS with the CRUD applications in mind. Thankfully the majority of web applications are CRUD applications. Let's take a look on what does Angular bring to the table:

- Routing – the routing in Angular is pretty similar to the server side routing allowing the application to change the URL. This means that loading a deep-linked page will not cause issues and the URL will update whenever you go into the site
- Templating – this is one of the key features of Angular. Not only that Angular supports templating but it also supports templating for displaying data in a handlebars-like manner like in the following code line: <h1>{{x1}}</h1> the content of the heading element will be the value in the variable x1.
- Data-binding and variable tracking – this is one of the biggest difficulties of all MV frameworks and Angular is pretty good at it. It is done automatically by simply adding variables to the scope object instead of manually declaring all variables as observables.
- It does not have event listeners – Functions can be declared and then bound to an event using the ng-click for instance. This is completely different from jQuery where the listener is removed from the template and declared independent of the view
- Good separation of concerns. Angular uses services, encapsulation, factories and dependency injection allowing more developers to work on the same project reusing code and keeping it DRY (Don't Repeat Yourself). [8]

3.7.2 Considerations while writing Angular code

Since Angular was built around the belief that declarative code is better when you have to build User Interfaces and wire software components together and the imperative one is better for expressing business logic

Here are some good ideas to keep in mind while writing AngularJS code:

- Decouple DOM manipulation from the app logic in order to increase testability
- Testing is as important as writing code since the testing difficulty is affected by the code structure
- Keep the client side of the application decoupled from the server side. This allows development work to be done in parallel on both sides

3.8 Node.js

Node.js and the NPM (Node Package Manager) were used for solving the dependencies of the project and for the integration of the needed packages in the project.

3.8.1 Introduction to Node.js

Node.js is a server-side platform build on Google's V8 Engine (JavaScript engine) and it was developed by Ryan Dahl in 2009.

Node.js is an open source cross-platform runtime environment for server side developing and networking applications. Since the Node.js applications are written in JavaScript they are compatible with Windows Linux and Mac OS X.

Some of the most important features of Node.JS:

- Event driven and Asynchronous – All the Node libraries are non-blocking because they are asynchronous. This essentially means that a Node server will never wait for an API to return data.
- Fast – Because it was built on the Google V8 JavaScript Engine, it is very fast in code execution
- Single Threaded but Highly Scalable – A single threaded model is used even while looping. The event mechanism helps the server to respond in a non-blocking way resulting in a highly scalable server as opposed to the traditional ones which had limited threads to handle the requests.
- No Buffering – The data is never buffered. The Node applications output the data in chunks

3.8.2 Node Package Manager (NPM)

The Node Package Manager (NPM) is the default package manager for Node.js. It consists of an online database of packages and a command line client.

NPM helps JavaScript developers share their solutions for particular problems for other developers to use them as well. These solutions are called packages and sometimes modules. A package is a directory with some files in it and also a file named "package.json" containing metadata about the respective package. A typical application will depend on a pretty large number of packages, sometimes up to hundreds, but the packages are often small.

The idea behind node packages is that a developer creates a small block of code that solves a particular problem. So using these block-like packages you can build more complex applications.

A broad variety of packages and modules can be found while browsing the NPM website. Since it was initially started as a package manager for the Node.js server applications you will find many modules to help on the server side but at this point there are a lot of packages for the front end use as well. [8]

3.8.3 The package.json file

Using a package.json file is probably the best way to manage your locally installed NPM packages. Here are some key features a package file has:

- It helps making your build reproducible and so it makes it way easier to be shared with other developers.
- Allows you to specify a version of the package you want to use in the current project.
- It serves as a documentation including which packages your application is depending on.

The package.json file is basically made up of JSON objects for each package all contained in a wrapper object. As a bare minimum the file should have:

- Name – the field must respect the following conventions: all the characters should be lowercase, the name should be only one word without spaces but dashes and underscored to delimit the name are allowed.
- Version – the version should be in the standard form of x.x.x in example 1.0.0

3.9 Bootstrap

Twitter bootstrap, also known as simply Bootstrap is an open-source framework for CSS HTML and JavaScript. Briefly after its launch it became the most popular project on GitHub.

One of the main reasons of Bootstrap being used so often is that it allows the building of aesthetic site and clean interfaces without putting much effort into it. Bootstrap also offers a wide range of free website templates and as well as components required for development.[5]

Using CSS frameworks and Bootstrap inside an ASP .NET MVC application works brilliant. Bootstrap manages the typography, form layout and the user interface components while allowing the developer to focus on the back-end functionality. This approach is important especially for small start-up companies which may not have a designer in their team. Starting with Bootstrap3 the mobile-first approach was implemented, which means that every site built on Bootstrap will be automatically optimized for small displays as well.

Some of the reasons that make Bootstrap a great framework for web design:

- Reusability: a modular template is preferred because there is no need of duplicating code for different sections of the design. Bootstrap already comes with pre-made components ready to be used.
- Consistency: the code readability is crucial for a designer. This trait helps when there is a larger number of developers on the same project. So that all of them get a better understanding of the code.
- Flexible grid layout: Bootstrap comes with an incorporated grid consisting on up to 12 columns growing with the screen resolution.

3.10 Gulp

Gulp is a tool usually used to perform repetitive tasks, especially on the front end part of an application. Some of the main reasons for which gulp is used in projects are the following:

- Reloading the browser when a file is saved in the project
- Optimizing assets (JavaScript, CSS, images) for production
- Spinning up a web server
- Using CSS preprocessors like Sass to compile SCSS files into CSS files.

Gulp is referred to as a build tool because it is a tool for running the tasks needed to build a web application. Gulp is one of the most used build tools right now along with Grunt.

Gulp can be installed using Node.js from the command line of the NPM package manager.

For our project Gulp was used to optimize the JavaScript, CSS files and images in order to get a faster load of the application and for compiling TypeScript into JavaScript to be processed by the browser.[9]

4 Detailed Design and Implementation

Following up some of the use cases in the application will be illustrated and explained

4.1 Use Cases

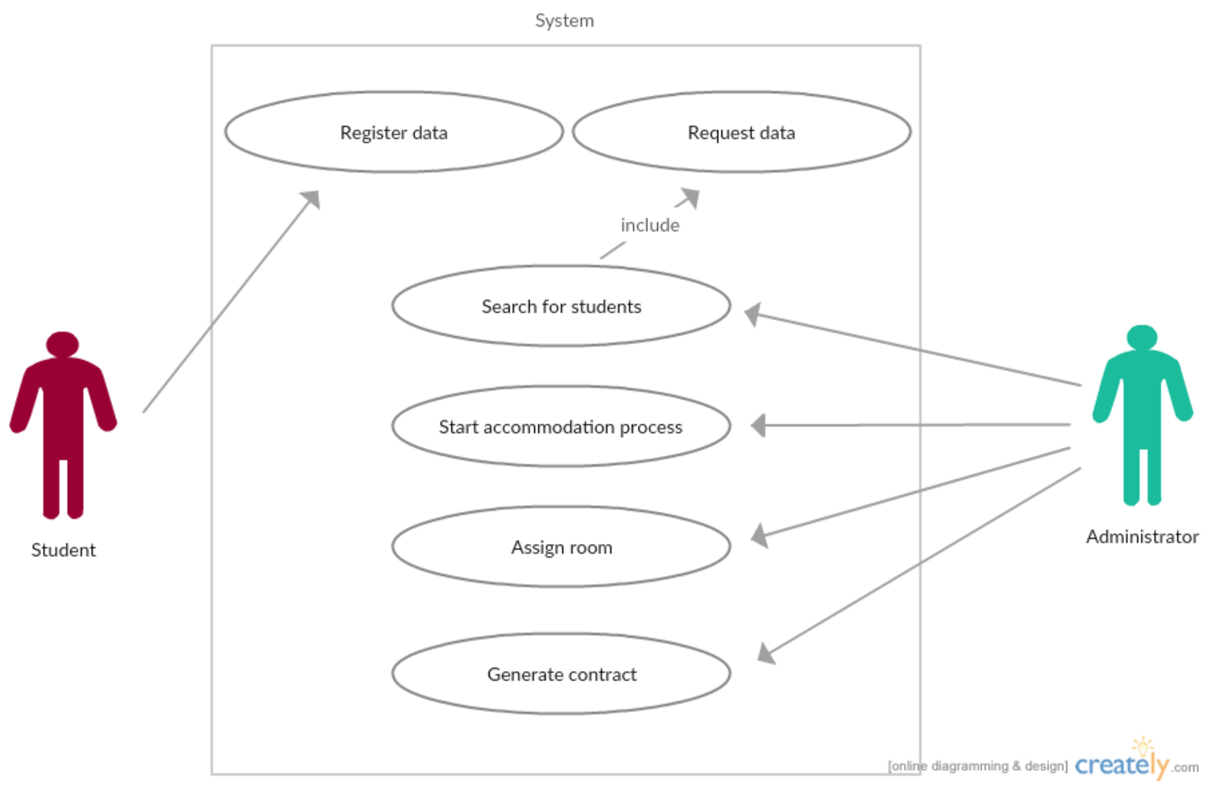


Figure 4.1 Use case diagram for the application

4.1.1 Student

The student scanning the QR code placed in front of the Dorm building will be redirected to the Student Register page in the Dorm Accommodation Application. After the respective page has opened up on his mobile device he will be allowed to complete a registration form with his personal information required for the dorm accommodation in order to ensure a faster process.

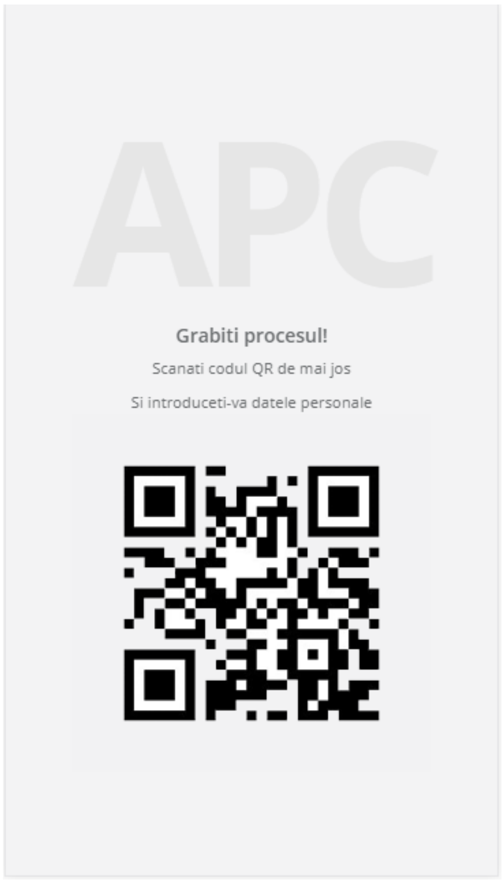

APC
Grabiti procesul!
Scanati codul QR de mai jos
Si introduceti-va datele personale

Figure 4.1.1 QR code poster in front of the dorm building

4.1.2 Administrator and committee member

The dorm administrator and the accommodation committee members each possesses a username and a password for logging into the application. Once they authenticated they can make the following actions.

4.1.2.1 Find and start the accommodation for a student

They can navigate to the Student Search page. On this page they can either select or search for the student they want to start the accommodation flow for.

4.1.2.2 Accommodate the student

The administrator or committee member can check if the personal information of the student is valid and edit it if it is not, select a floor and after that a room on that floor in which the student will be accommodated. After checking the data and selecting the room they can add a special case to the respective student in order for him to have a monthly payment discount, and then generate the custom contract for the student.

4.1.2.3 Check the dorm status

The user can access the dorm's status page seeing each floor with their rooms and the number of occupied and open places. The rooms are colored differently depending on the number of open places in that room. They can select any particular room and view its details (total places, available places and students currently in that room). They can also choose to check out a student from the room they are viewing the details of.

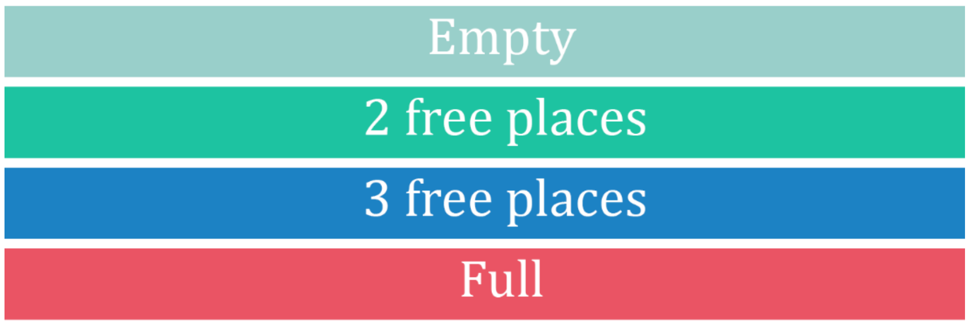


Figure 4.1.2.3 Room status color palette

4.2 System Architecture

The system consists of two principal components interacting with each other: the web application and the server. Both of these communicate with each other when the administrator or committee member does any of the previously mentioned actions or when a student enters the register page and adds his personal information.

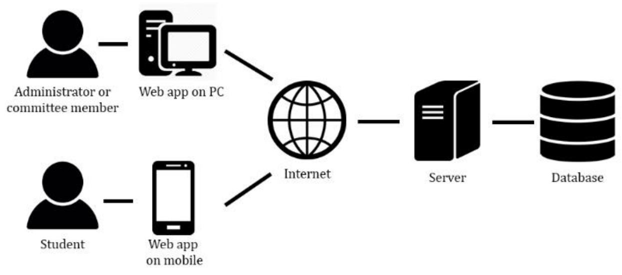


Figure 4.2 System Architecture

For the web application to communicate with the server a request has to be made to the Web API Controller which depending on the request will send the corresponding content

The majority of server requests are for accessing or modifying various data in the database

4.3 Database Structure

For the creation of the database MS SQL Server 2014 was used. The current SQL server was chosen because it is free and it allow the usage of the SQL Management Studio which comes with a user-friendly interface in order to manage the databases. Another reason for which this server was used is because it's a Microsoft product so one does not have to install additional tools in Visual Studio 2015 in order to start a connection to a database

The database consists of 11 tables soon to be explained. A diagram of the current database can be seen in the following figure.

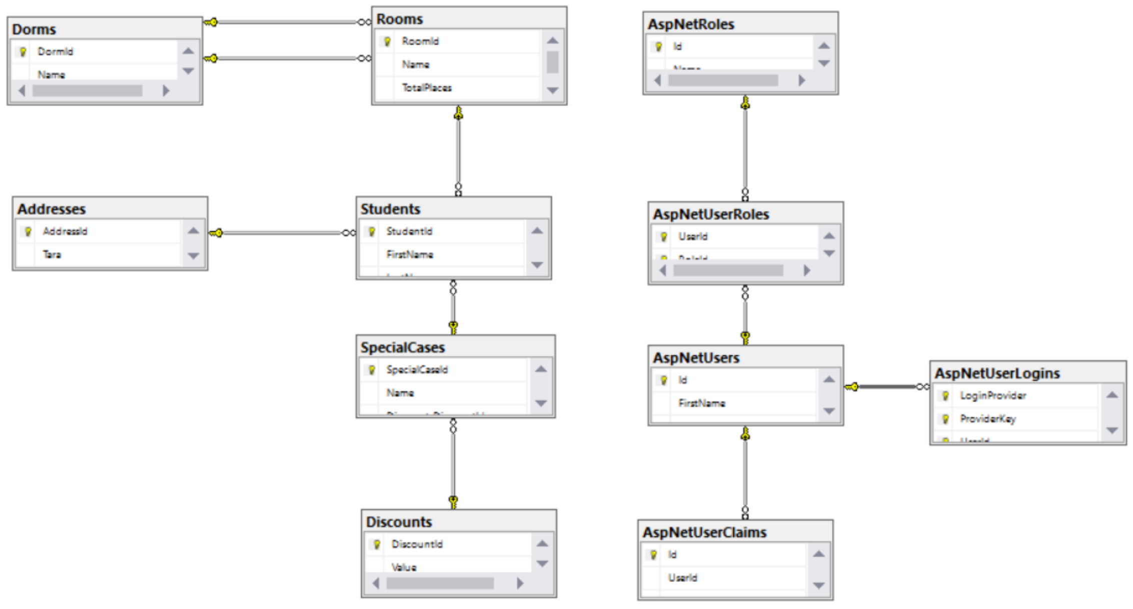


Figure 4.3 Database diagram

The Students table consists of personal information about the student, a foreign key to a Room, an Address and a Special Case if one was defined.

For the management of the Students, the Students table was created and it contains personal information about the students. Three foreign key for AddressId, SpecialCaseId, RoomId to link the student to a specific address, special case and room in the database. In The SpecialCases table the special case names and the foreign key DiscountId in order to link the special case to the required discount from the Discounts table.

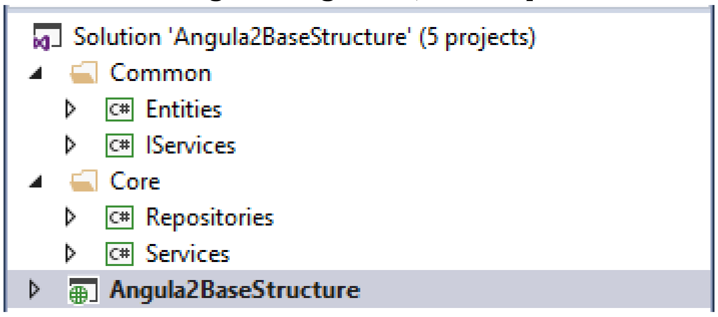
The Dorm table contains an Id and a Name for each dorm and foreign keys for RoomId for each room that can be found in that particular dorm.

The remaining tables AspNetRoles, AspNetUserClaims, AspNetUserLogins, AspNetUserRoles and AspNetUsers are used by the authentication service implemented in the application. The AspNetUserRoles table contains foreign keys to the UserId and the role each user has (the role can be either administrator or committee member) while the AspNetUsers contain the user name for each account and the password hash key.

4.4 Web Application

The web application consists of the following five projects:

- Entities – Containing the entities used by the application
- IServices – Containing the used services' declarations, in example logging, mapping, repository
- Repositories – Implements the services declared in the Repositories folder of IServices. This project contains the Entity Framework part of the project which links to the database on the server.
- Services – Contains the rest of the services' implementations
- Angular2BaseStructure – This is the main project where the backend is created using the ASP .NET controllers and the frontend where the web pages and the UI are created using the Angular2 JavaScript framework.



```
Solution 'Angula2BaseStructure' (5 projects)
  Common
    Entities
    IServices
  Core
    Repositories
    Services
  Angula2BaseStructure
```

Figure 4.4 Solution structure

For the web application the MVC pattern was used consisting of three main components: the controllers which act when the user interacts with the application, the models that handle the database access part and the views used to display the data requested by the user.

Since the Angular2BaseStructure is the main project folder it will contain all the setup data required for the project to compile and link to the required resources in order for the app to become functionally.

Following up a few files in the App_Start directory will be explained.

The BundleConfig.cs file links the plug-in bundles. The main two categories are the scripts and styles bundles. The styles bundle links all the CSS files in the project such as the bootstrap ones, the theme CSS files and the custom CSS of the application. If one desires to add some third party plugins their CSS files should be as well included here using their relative path in the project structure tree. For the JavaScript files the process is exactly the same as for the CSS files.

The IdentityConfig.cs file handles the authentication/registration constrains. In this file one can find the password validation criteria as well as the error handling in the case when the entered password does not match the required criteria

Going into the Infrastructure/Controllers folder the abstract class defining the WebApiController can be found. Every single controller in this application extends this class.

4.4.1 Implementation of the dependency injection

For the dependency injection the Simple Injector library for dependency injection in .Net 4 or higher. The goal of this library is to provide the developers with a simple and fast Inversion of Control method that promotes best practice. The general idea behind the Simple Injector, in fact the idea behind the Dependency Injection principle is that you can design your application around loosely coupled components using the Dependency Injection pattern.

In the ContainerConfig class the Dependency Resolver was instanced to be a new instance of the Simple Injector Web API Dependency Resolver.

The _ApiUrls.cshtml file is used for simplifying the routes for the API controllers and methods by linking each API URL to a list item. Therefore when an API call is made on the frontend part of the application, the system will navigate to the _ApiUrls file and select the required URL so that the requests ends up on the right controller.

The _Layout.cshtml file contains the skeleton of the html page defining the DOCTYPE and metas. It also includes a script tag with the bootstrapper. This page handles the partial rendering of the view using the components in the Scripts folder. The bootstrap.ts file imports the main component (AppComponent), the LocationStrategy and the providers and bootstraps them together.

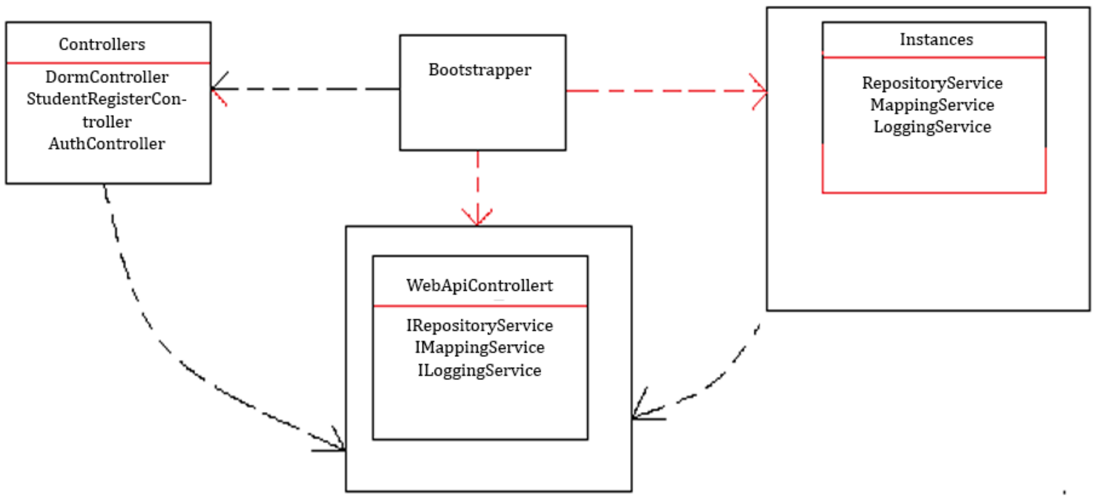


Figure 4.4.1 Dependency Injection Example

4.4.2 Entity Framework implementation

For the database creation and communication, the code first Entity Framework was used. Code first means that you do not design the database structure using tables and relationships but create the Entity files for each table with their references and then using the NuGet Package Manager command line update the database to the current status of the entities. This usually ends up in the automatic creation of a new Migration file that contains the SQL code that needs to be executed in a query on the database in order to update it to the current status.

For the web application the automatic migration creation was disabled because it is more practical to save the Migrations by hand, only when the developer thinks that a migration is required and by doing this it also allows a faster rollback to a previous stable version of the database in case that something goes wrong while updating it to the newest migration.

The environment will not allow us to update the database if there are some changes in the entities that are not saved in a migration and it will throw an error message suggesting that the creation of a new migration is needed before updating the database.

Each migration should be saved using a representative name for what modifications have been made, using camelCase (linked words with no space each beginning with Uppercase). An example of migration is AddedStudentAddress.

4.4.3 Bootstrap template implementation

During the application's development, on the front-end part the CSS and JavaScript framework known as Bootstrap was chosen. Bootstrap was used because it is fast and simple to use even by developers without much experience with CSS and JavaScript.

For the application the free admin dashboard template named Inspinia was used. In order to use the above mentioned template the required CSS and JavaScript files needed to be downloaded in order to use the full range of functionalities offered by the Inspinia template. Using a template will result in a sharp user-friendly interface.

Using the Bootstrap framework allow the further use of modals allowing a fast and easy pop-up windows creation in case the users want to make an action that has no return option like checking out a student from the room he is accommodated in.

The Bootstrap theme also offers a broad variety of CSS classes for quick table, forms, menus creation and so on. The main benefits of this theme is that first of all, it is free, comes with great responsiveness and it is really easy to use even by beginner developers.

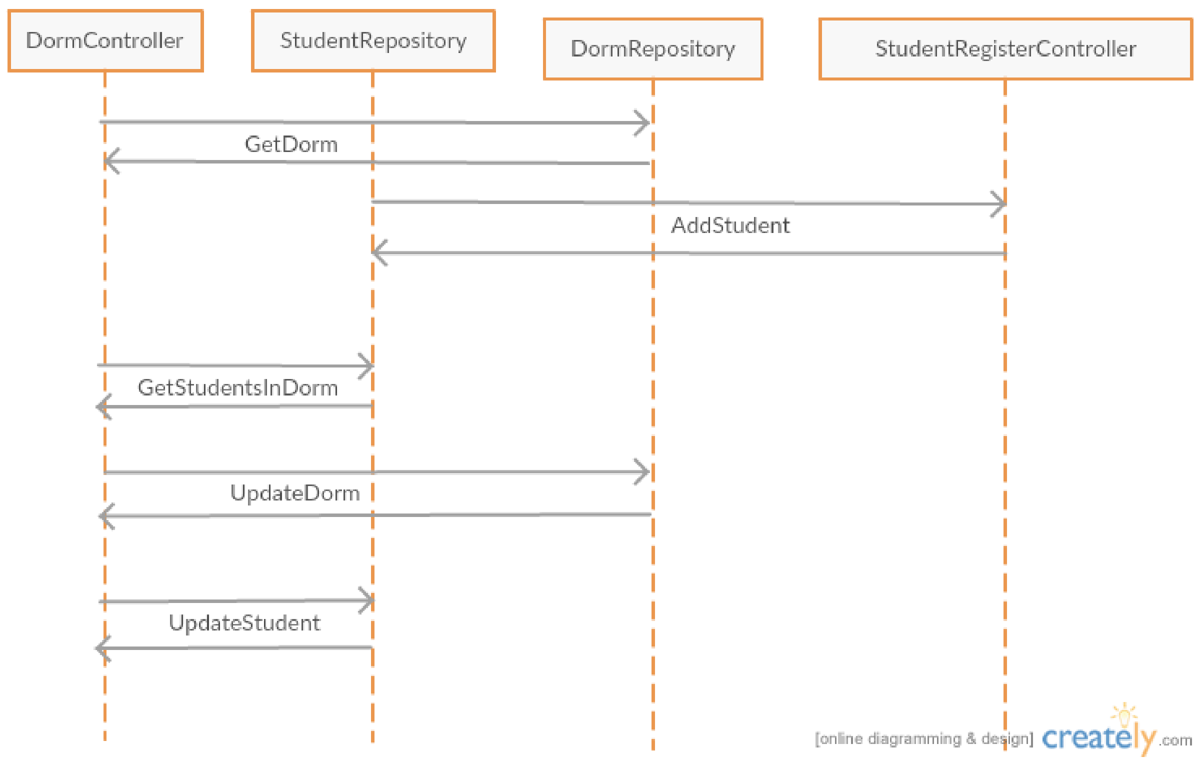


Figure 4.4.2 Register and accommodate student sequence diagram

4.4.4 WEB API application implementation details

In order to implement the web application the following controllers were required: AuthController, DormController, StudentRegisterController.

For the database access and data manipulation Entity Framework was used. This framework was chosen because it is easy to use and maps the database tables in easy to manipulate objects.

In order to operate on the database, the connection to the database is created and after that the Entity type objects generated by the Entity Framework are used. When the CRUD (create, read, update, delete) operations are called on the database, a connection to the database will be created and the table mapped to the required object would be selected and the used operation will be made. Entity Framework helps mapping foreign key references so that if the inclusion of data found in another table is required the only thing that needs to be used is the Include ("NavigationProperty") method has to be used where the NavigationProprety is the table one need to include. When mapping the tables in the database a diagram with the mapped entities and their relationships will be created so that a graphical representation of the database will be available.

Following up, the StudentRegisterController will be briefly explained. The HTTP Post AddStudent method receives an object of the Student Entity type as a parameter and inserts all the data in that object into the Students table of the database, as well as into the Address table of the database and creates the required foreign key between the student and the address. The HTTP Put UpdateStudent method also receives an Student Entity object as a parameter but this time it queries the database by the received StudentId in order to find the respective student, then updates all the columns for that student that have changed.

The DormController contains a HTTP Post method named CazeazaStudent used for assigning the respective student to the desired room. It receives as a parameter an object made up of a DormId a RoomId and a StudentId. The method first finds the dorm, the room and the student with the respective ids in the database and then creates foreign keys in order to link the student to the room and add him into the StudentsInRoom column in the Rooms table of the database. Since this controller has the ability to accommodate a student into a specific room it also has a method for checking out the student from that room. It receives the same object as parameter like the CazeazaStudent method but it does the exact opposite: finds the dorm student and room, removes the student from the respective room and then updates the tables and foreign keys as well.

The Business Logic directory contains a class used to generate the contract for a student. The class' name is DocumentGenerator and the GenerateDocument method inside of it receives a Student object as a parameter. It replaces the placeholders in the contract document template with the student's information and saves it in a file named using the following convention: CONTRACT_STUDENT-FIRST-NAME_STUDENT-LAST-NAME and saves it on the server. The naming convention is helpful when searching for the contract generated for a specific student in the Contracts directory on the server. If a contract for the current student can already be found on the server, the DocumentGenerator will replace it with the new generated contract. In this case there will not be duplicate contracts for the same student on the server.

For the authentication functionality the API requests are handled by the AuthController. The register method is the only API call in the application that allows anonymous access (it does not require to have an authenticated user active). When registering a new account the basic account data is required (username, e-mail, first name and last name) and a password. The password has to match a few criteria in order to be validated and for the account to be created. The criteria for a password to be valid are the following:

- Minimum length is 6 characters
- At least one digit or non-letter character
- At least one lowercase character
- At least one uppercase character

For the authentication the Microsoft OWIN library was used and the password string entered by the user when registering an account will be encrypted using the SHA encryption. Therefore in the database for the user entity only the generated password hash will be saved instead of the password string. When the user logs in, the password he will type in will be encrypted with SHA as well and only if the current hash matches with the one in the database the authentication will be valid.

In the Configuration.cs file a seed method has been implemented to run every time the "update-database" command is ran in the Nuget Package Manager Console. The seed method then calls up on a Dorm seed method that automatically generates all the rooms in a dorm with all their details (room number, number of places, occupied places which is 0 by default). This functionality can come in really handy in the case of a server error which leads in the loss of the database and so of the dorm data.

4.4.5 Front-end implementation

Since the front-end part of the application was developed using Angular2 it doesn't consist in a series of web pages but in a number of components and partial views each being rendered under certain conditions and only when required.

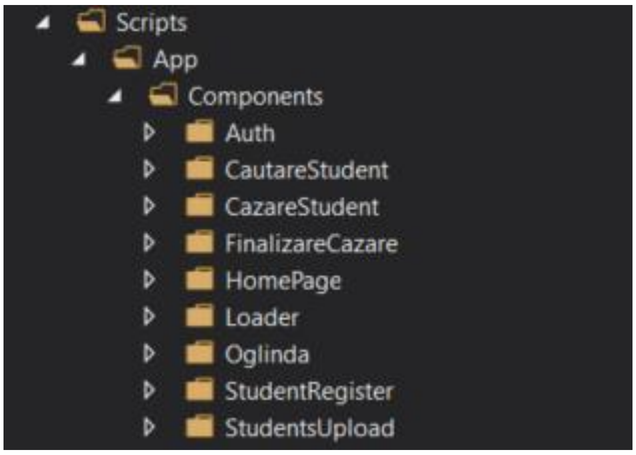


Figure 4.4.3 Components tree structure

Each directory in the figure above contains a component file (example.component.ts) and a template file (example.template.html). The component file includes all the frontend functionality such as functions and API calls through the HTTP service included in Angular.

The Auth directory contains the Login and Register components. For each one of them the template contains a form and a submit button. The Register component takes the values bound to the input fields and creates a JSON object with them and then passes them to the AuthController in order for the account to be created as mentioned in the previous paragraphs. The Login component does mostly the same but the API call is different and it only sends the user and the password string to be encrypted and compared to the hash found in the database for the entered username. If the authentication is not valid an error modal will pop up telling the user that the password or the username is incorrect, otherwise the user will be redirected to the first view in the application.

In the CautareStudent component the table containing all the students in the database and the functionality for it can be found. Each student has a button with the option to start the accommodation process. If this process is started the application navigates to the CazareStudent component by passing the student's data as a JSON string through the URL which will be parsed into a JSON object on the CazareStudent view. After the desired room has been picked, the student JSON object will be updated and an API call will be made by the HTTP service in order to update the database. Clicking the finish accommodation button will convert the student entity back into a JSON string and pass it through the URL to the FinalizareCazare component where it will again be parsed back into the JSON object. If the case requires the student can receive a discount for the monthly payment and the option to generate the final contract for that student. When generating the contract the student object will be passed to a DocumentGenerator class instance in the controller through a HTTP Post request where the contract will be filled with his personal information and then saved on the server.

The Oglinda directory contains the component representing the mirror of the dorm. Here one can find an overview of the dorm showing each floor with its rooms in suggestive colors. When this view initializes (on the ngOnInit() method) an HTTP get request will be sent to the DormController returning the entire Dorm Entity from the database. After the response from the API has arrived the dorm entity's lists for rooms on each floor are turned into particular tiles for each room using one of the best features of Angular, which is the ngFor (ngFor is basically a for loop for generating repetitive HTML code on template load). Clicking on any room will result in a modal containing information about the room showing up. In this modal one will find details like total number of places and total available places in that room and the list of students already in that room.

On the front-end the HTTP requests are performed by a custom HTTP service named ApiService. The service sets the request header content type for each CRUD operation to the JSON format, appends the authorization bearer token to the headers and returns the response of the requests using promises and observables.

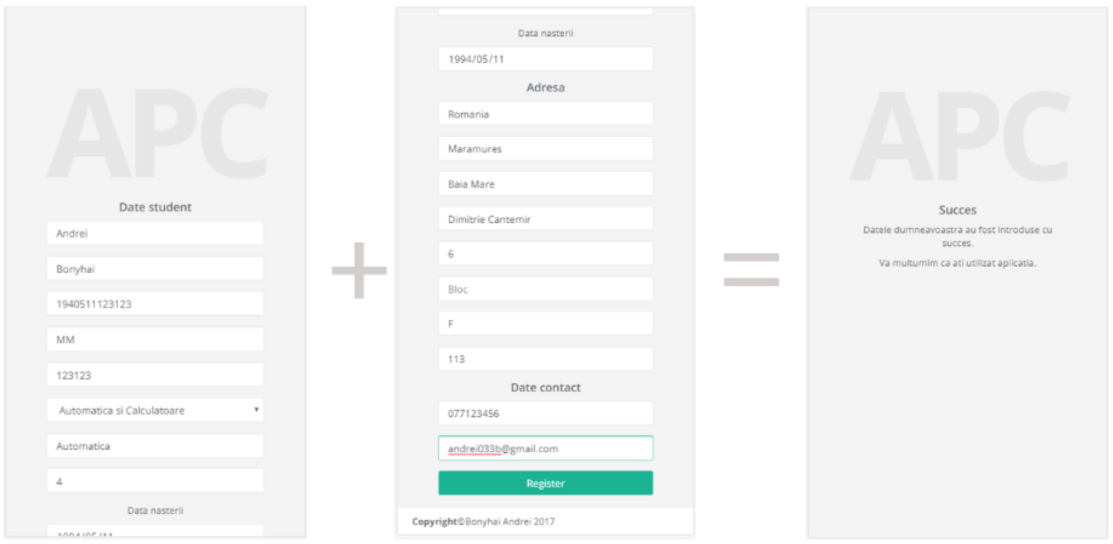
The main component for the application is the app.component file. This component imports all of the other components and glues them together. It mainly consists of the sidebar navigation menu and a container in which the partial views will be rendered depending on the accessed route. The application router is implemented in this component and the other controllers are bound to application routes.

5 Testing and Validation

During the testing the application had to be tested as well as from a small resolution browser like the ones on a phone without authentication required and from a standard browser with authentication.

The testing will be performed locally using Visual Studio 2015 to run the web app and the Google Chrome browser to access the application.

The first testing case will be the one of the student scanning the QR code in front of the dorm building and then registering his data into the registration form and then sending it to the API to be processed.



Andrei
Bonyhai
1940511123123
MM
123123
Automatica si Calculatoare
Automatica
4
Data nasterii
1994/05/11

Data nasterii
1994/05/11
Romania
Maramures
Baia Mare
Dimitrie Cantemir
6
Bloc
F
113
077123456
andrei033b@gmail.com

Copyright©Bonyhai Andrei 2017

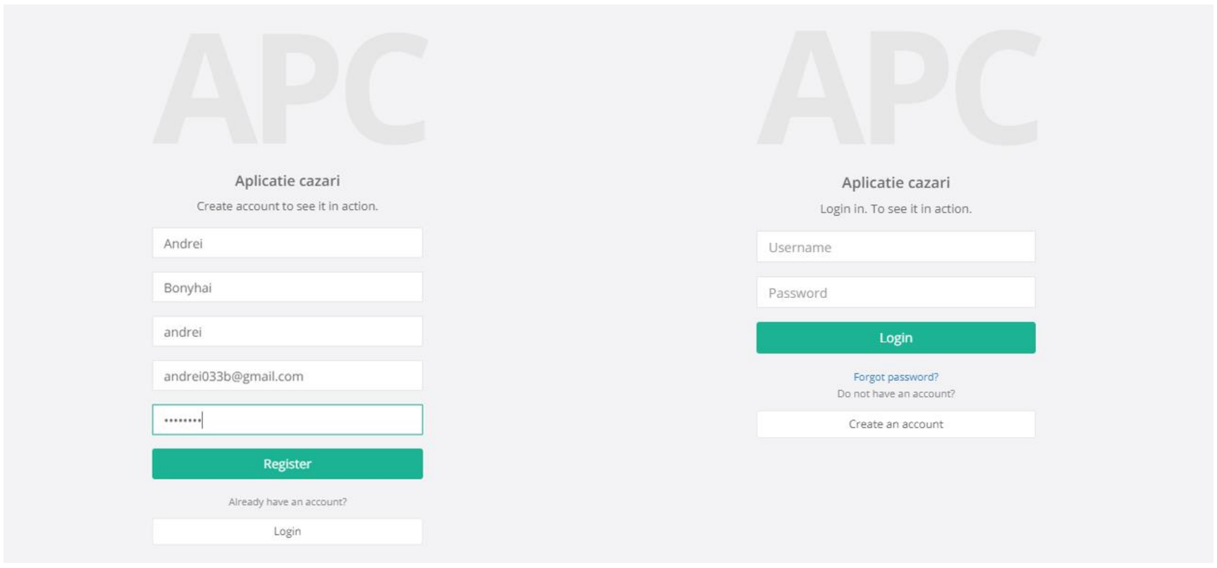
Datele dumneavoastra au fost introduse cu succes.

Va multumim ca ati utilizat aplicatia.

Figure 5.1 Mobile browser register flow

One can see that the data was registered successfully and the user was notified by receiving a success screen on his mobile device. The registration form has a smooth flow on a mobile display thanks to the Bootstrap theme used to develop the application.

The second testing case will be the one when the administrator or committee member registers an account then logs in with the created account. For the registration the user will be required to enter his first and last name, e-mail, select a username and a password that matches all the criteria described above in the AuthController description. After the register button will be pressed and if the information is correct the user will be automatically redirected to the login page.



Andrei

Bonyhai

andrei

andrei033b@gmail.com

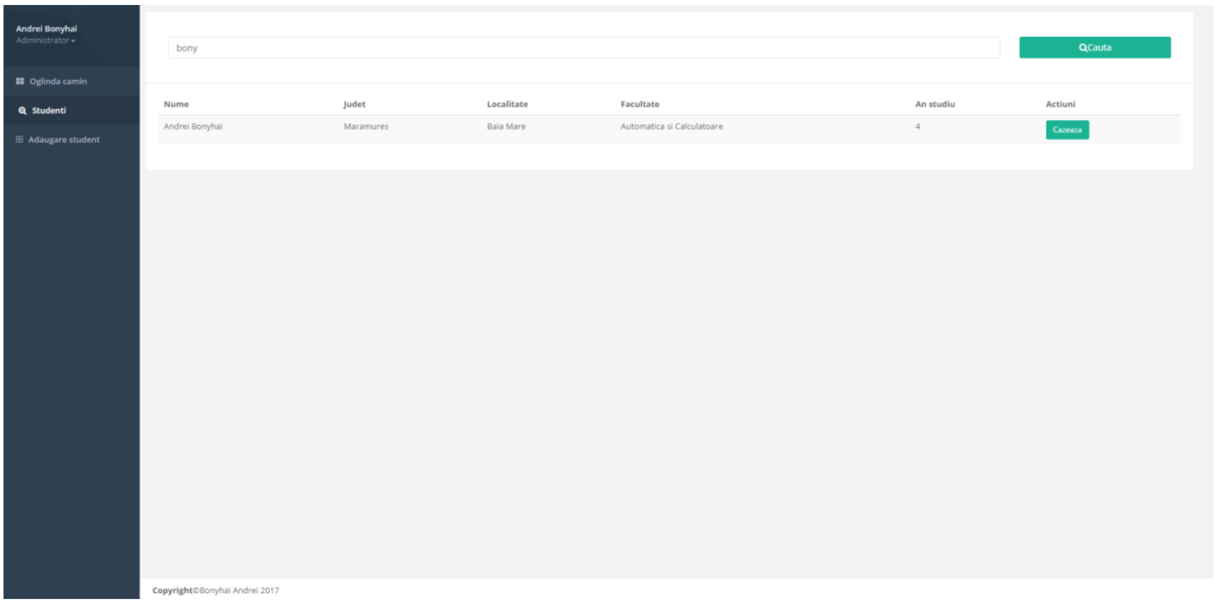
••••••••

Username

Password

Figure 5.2 Register and Login screens

The next testing case is when the administrator or a committee member is searching for a student in the database and the table is populated only with the students which names contain the characters entered in the search bar before pressing the search button. If the search button is pressed without any characters entered in the search bar the complete list of students will be returned.



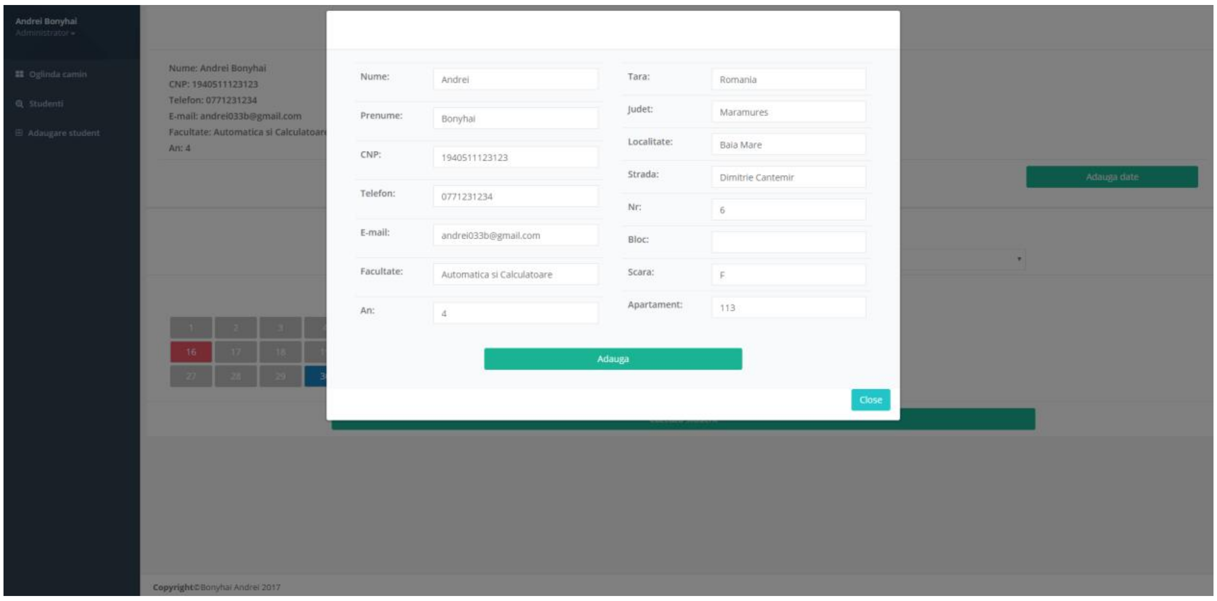
bony

Nume	Judet	Localitate	Facultate	An studiu	Actiuni
Andrei Bonyhai	Maramures	Baia Mare	Automatica si Calculatoare	4	Cazeaza

Copyright©Bonyhai Andrei 2017

Figure 5.3 Student search screen

The fourth case is when the desired student was found in the table and the accommodation process has begun. The administrator or the committee member will be able to check the student’s personal information, will select the floor and then the room in which the student will be accommodated. If the user choses to verify and add some additional data for the student the Add Data modal will pop up and he will be able to edit all the information in the specified input fields and then update the data into the Students table of the database. The Add data modal is presented in the figure below.

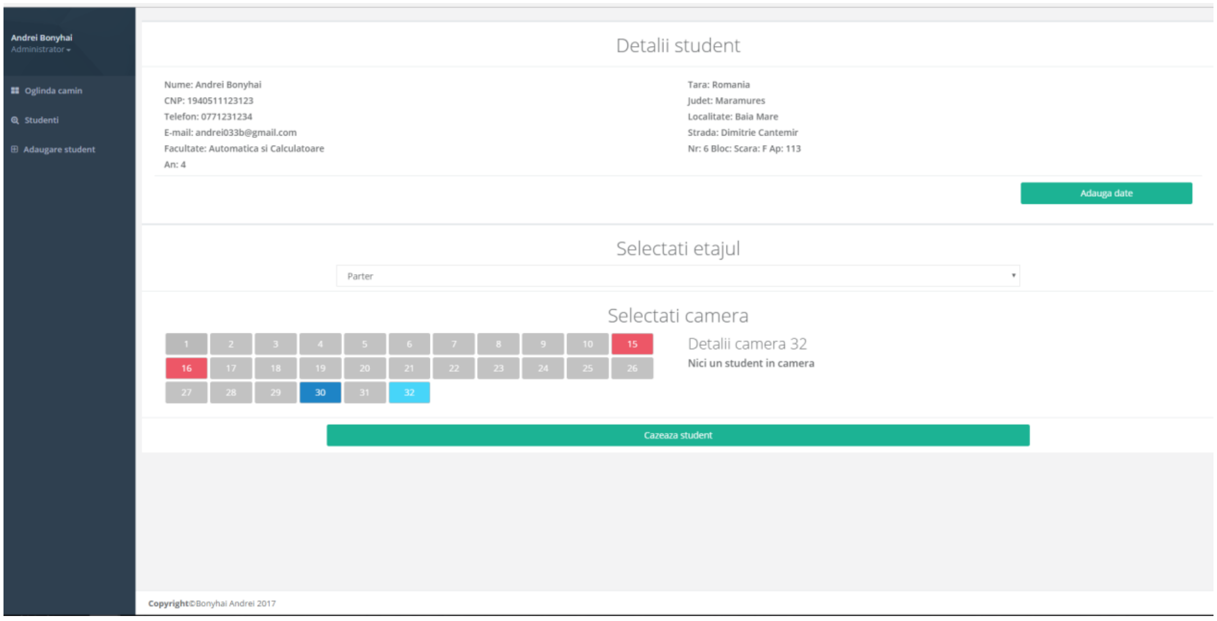


Nume: Andrei Bonyhai
CNP: 1940511123123
Telefon: 0771231234
E-mail: andrei033b@gmail.com
Facultate: Automatica si Calculatoare
An: 4

Nume:	Andrei	Tara:	Romania
Prenume:	Bonyhai	Judet:	Maramures
CNP:	1940511123123	Localitate:	Baia Mare
Telefon:	0771231234	Strada:	Dimitrie Cantemir
E-mail:	andrei033b@gmail.com	Nr:	6
Facultate:	Automatica si Calculatoare	Bloc:	
An:	4	Scara:	F
		Apartament:	113

Figure 5.4 Student Add Data modal

If the student's personal information required for the accommodation is complete, a floor will be selected from a drop down menu. The room tiles will then be rendered according to the selected flor and one of them will be selected and highlighted as selected. Next to the room tiles the details for the selected room will be displayed as shown in the figure.



Detalii student

Nume: Andrei Bonyhai
CNP: 1940511123123
Telefon: 0771231234
E-mail: andrei033b@gmail.com
Facultate: Automatica si Calculatoare
An: 4

Tara: Romania
Judet: Maramures
Localitate: Baia Mare
Strada: Dimitrie Cantemir
Nr: 6 Bloc: Scara: F Ap: 113

Selectati etajul

Parter

Selectati camera

Detalii camera 32
Nici un student in camera

Figure 5.5 Selected room for the student

After the "Cazeaza student" button is pressed the current student will be added to the selected room's students in room list and the next view for the accommodation process finale will be rendered. On this view the administrator or committee member will be able to add the discount previously mentioned by selecting the special case of the student in case. In example if the student is an orphan he will get 100% discount and if the student is a committee member he will receive a 25% discount to the monthly rent.

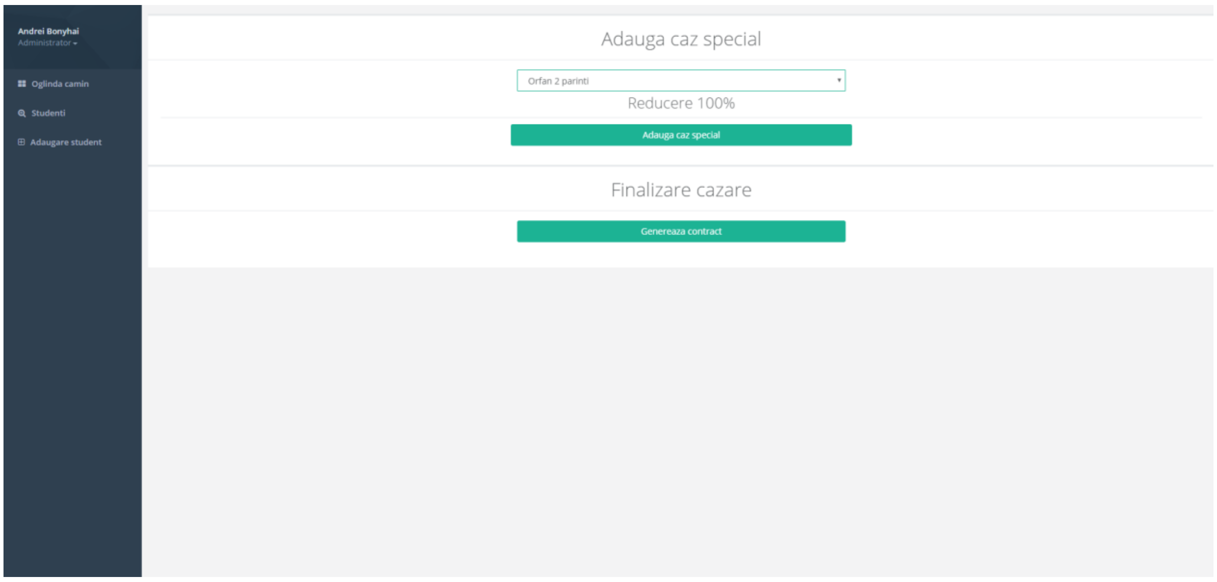


Figure 5.6 Special case setting and contract generation

When pressing the Generate Contract button, the application will automatically fill out the accommodation contract for the student in case and save it in the Contracts folder of the server.

The final tested case was the one of the dorm's overview. On the top of the screen statistics about open and occupied places are shown and the dorm's room tiles are shown and colored according to the number of free places left in that room.

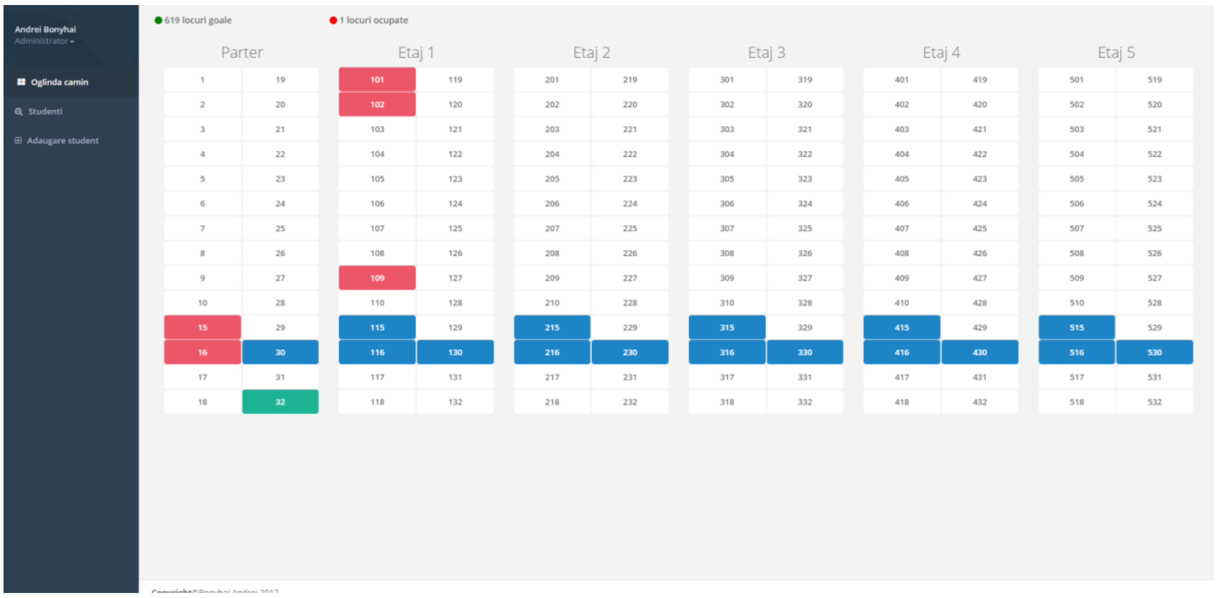
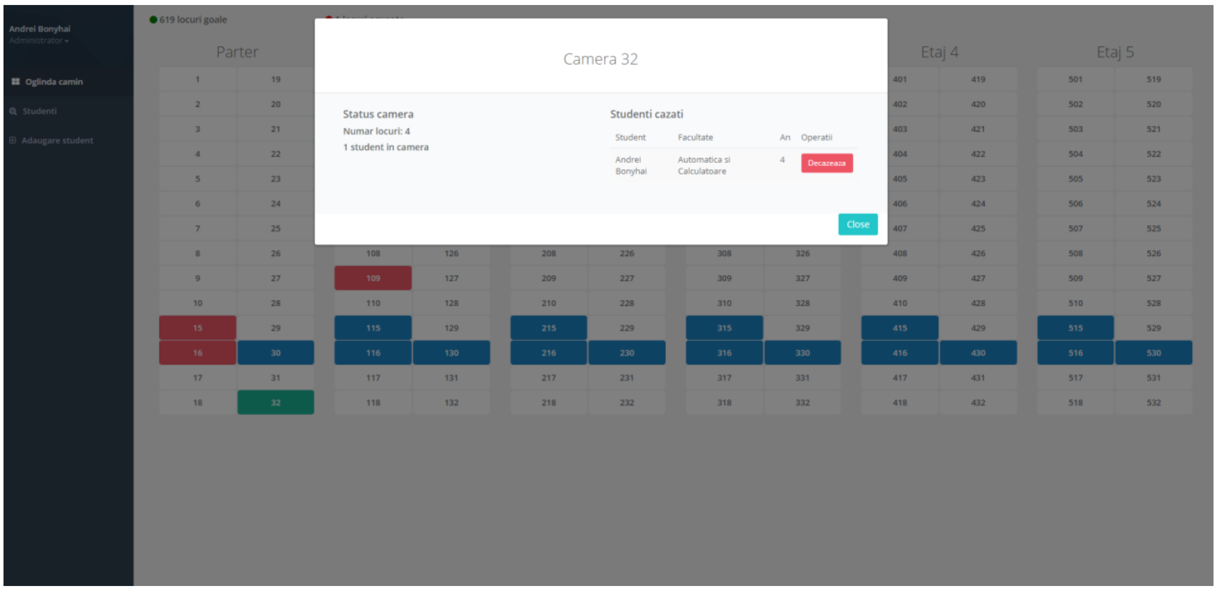


Figure 5.7 Dorm overview screen

When a room tile is clicked the room modal for the respective room will pop up showing all the information about that room as shown in the following figure.



Camera 32

Status camera
Numar locuri: 4
1 student in camera

Studenti cazati

Student	Facultate	An	Operatii
Andrei Bonyhai	Automatica si Calculatoare	4	Decazeaza

Figure 5.8 Dorm overview room modal

From this modal the administrator and the committee members can check out a student if he does not want to live in the dorm any more or if there was a human error and he ended up accommodated in the wrong room.

6 Installation guide

In order to run the application locally on a computer the following steps have to be taken:

- Install Visual Studio 2015
- Install Microsoft SQL Server 2013 or newer
- Install the .NET Framework
- Install Node.js
- Navigate to the ../Angular2BaseStructure/Angular2BaseStructure folder
- Shift+ Right Click inside the folder explorer and open a Command Prompt window
- In the Command Prompt run the *npm install* command in order to install all the packages required in the package.json file using the NPM package manager. Then the node_modules directory will be automatically created and it will contain all the packages found in the package.json file installed in it.
- Open the Angular2BaseStructure.sln solution in Visual Studio and then put the program in Run mode
- Open a browser and navigate to localhost on port 1234 (localhost:1234) in order to access the application

For the application to run, the computer must have at least 1 GHz processing power and 512MB of RAM memory. The computer must have installed IIS version 7.5 or newer.

7 Conclusions

For the development of the Student Dorm Accommodation Application a few libraries and frameworks that take over a part of the projecting and functionality tasks were used.

By developing the web application the speed of the accommodation process for each student was drastically increased since the personal information did not have to be entered by hand by the administrator and so the main bottle neck of the process was avoided. The QR code allowing students to register themselves also helps the speed and it gives the students the feeling that they are participating in the accommodation process as well, more than just waiting in line and then waiting for the committee to place them in a room.

7.1 Further Development

The application is currently designed to support only one dorm from the Observator campus of the Technical University of Cluj-Napoca. It can easily be extended to cover all the dorms from this campus without too much work being required.

After being extended over the entire Observator campus it can be extended to cover all the dorms belonging to the Technical University and even dorms from other universities in Cluj-Napoca.

One ambitious expansion plan is to extend the application to cover all the campuses in Romania and making it become a standard application used by all the Romanian universities when it comes to the dorm accommodation for students.

Apart from the scaling of the application to national scale, user roles for students can be added so that all the students currently living in dorms would have an account on the application. Accomplishing this will allow a further extension of the application by creating a washing machine appointment feature so that there will be no overlapping washing machine appointments or students washing their clothes on other floors occupying the washing machines destined for the respective floor. This could even end up in students making washing machine appointments in different dorms in case the washers in their dorms have malfunctioned and there are only a few ones left.

Since all the students in the campus will own an account on the application, getting an 1 hour football reservation at the field in campus should become way easier because there will be no need of going to the campus cash desk in order to make the reservation and then get a receipt proving that you should be granted access to the football field. Another useful feature for the application could be an appointment system for the medical cabinet located in the campus. The students could make appointments and describe their symptoms before even going to the doctor. This way the medic would get an idea about what their illness is and could even respond to their appointment suggesting medication until the time of the appointment.

The most important extension feature would be the implementation of a payment functionality. In this functionality the students will be able to link a credit card to their account and make various payments regarding their staying in the dorm, such as paying the monthly rent and some other not as important payments such as football reservations payment.

Bibliography

[1] Roebuck, Kevin. *QR Code: High-impact Strategies: What You Need to Know: Definitions, Adoptions, Impact, Benefits, Maturity, Vendors*. Place of Publication Not Identified: Publisher Not Identified, 2011.

[2] Surhone, Lambert M. *QR Code: Barcode, Denso, Japan, Mobile Phone, Mobile Tagging, Object Hyperlinking, Open Format*. Beau Bassin: Betascript Publ., 2009.

[3] Galloway, Jon. *Professional ASP.NET MVC 4*. Hoboken, NJ: Wiley, 2012.

[4] Kanjilal, Joydip. *ASP.NET Web API: Build RESTful Web Applications and Services on the .NET Framework: Master ASP.NET Web API Using .NET Framework 4.5 and Visual Studio 2013*. Birmingham: Packt Pub., 2013.

[5] Shenoy, Aravind, and Ulrich Sossou. *Learning Bootstrap: Unearth the Potential of Boostrap to Create Responsive Web Pages Using Modern Techniques*. Birmingham, England: Packt, 2014.

[6] Jon Preece. *TypeScript Beginners Guide,* article from October 2015 on http://www.developerhandbook.com/typescript/typescript-beginners-guide/

[7] *AngularJS Documentation Developer's Guide* on the AngularJS Docs website https://docs.angularjs.org/guide/introduction

[8] *Introductive tutorial on Node.js* on the Tutorialspoint tutorials website https://www.tutorialspoint.com/nodejs/nodejs_introduction.html

[9] Zell Liew. *Gulp for beginners.* Article on the css-tricks webpage from September 2015 https://css-tricks.com/gulp-for-beginners/

[10] *ECMAScript 2015 language specification,* from the ECMA International website http://www.ecma-international.org/ecma-262/6.0/ECMA-262.pdf

[11] *Simple Injector library documentation,* on the Simple Injector library website https://simpleinjector.readthedocs.io/en/latest/quickstart.html

Figure 2.1 *QR code structure and explaining* http://www.lifeofanarchitect.com/wp-content/uploads/2011/05/Anatomy-of-QR-Code.jpg

Figure 3.1 *MVC Architecture structure and relationships*
http://www.joshholmes.com/blog/content/binary/WindowsLiveWriter/BuildingaSimplePhotoGall.NETMVCFramework_113D2/image_6.png

Figure 3.2 *ASP .NET Web API structure and pipeline illustration.*
http://dotnetmentors.com/Images/asp-net-web-api-request-pipeline.png

Figure 3.3 *Entity Framework, framework structure and architecture*
http://www.entityframeworktutorial.net/Images/ef-architecture.PNG