UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ
SPECIALIZAREA Informatica

LUCRARE DE LICENŢĂ

Mood Evaluation App

Conducător ştiinţific
Lect. Joszef-Árthur Molnar

Absolvent
Petruta George

2022

ABSTRACT

There is a rising concern about increasing rates of anxiety and depression among children and young people. Mental health problems cause major distress and a harmful impact on social relationships, school or even physical health and even though evidence-based interventions are available everywhere, in Europe and the USA, 50% - 78% of people with mental health disorders do not receive treatment.

Digital surveillance of symptoms offers a promising and innovative way to deliver interventions for depression and psychotic disorders. Therefore, mobile health (mHealth) offers an incredibly powerful platform for monitoring and management of mental health symptoms.

Advantages of mHealth include constant availability, lower cost, greater access and increasing service capability and efficiency.

Despite a large number of currently available apps, a 2013 review of literature highlighted how research has lagged behind in the field of app development. Despite some reported drawback of digital engagement, there is evidence for the positive impact of digital devices in peoples' lives.

Mobile phone use has almost achieved complete penetration, with 96% of the global adult population having a mobile phone subscription.

Cuprins

1 Introduction **1**

2 Theoretical part **2**
2.0.1 Motivation 2
2.0.2 Applicability 3
2.0.3 Description 4

3 Technical Part **5**
3.1 Front-end technologies 5
3.1.1 Angular 5
3.1.2 Ionic 6
3.2 Back-end technologies 6
3.2.1 ASP.NET CORE 6
3.2.2 Database 10
3.2.3 HyperText Transfer Protocol (HTTP) 10

4 Architecure of the app **12**
4.0.1 Features 12
4.0.2 Login 14
4.0.3 User Management 16
4.0.4 Domain 18

5 Conclusions **19**

Bibliografie **20**

Capitolul 1

Introduction

Capitolul 2

Theoretical part

2.0.1 Motivation

Depression and anxiety are considered as 'common mental disorders' that are denoted as such because of their high prevalence among the general population (WHO, 2017). Depression is defined as "a common mental condition that causes people to experience low mood, lack of interest or pleasure, feelings of guilt or low self-worth, disturbed sleep or eating, low energy, and poor concentration," according to the Mental Health Foundation (2018a). Anxiety is defined as "a continuous feeling of unease, concern, or fear that impacts daily living and may be an indication of an anxiety disorder" (Mental Health Foundation, 2018b). The causes for mental health issues may vary, from childhood trauma to unhealthy workplace or even bad relationships.

Nowadays, the need for mobile apps has increased significantly, because the mHealth industry has developed more and more, leading to an on-growing transition from traditional interventions to digitally delivered ones. Surveys show that the benefits of on-line interventions are more efficient than the conventional ones, because it is a way more accessible approach and the cost of implementation is reduced.

Some of the reasons people refuse to get help and seek therapy is the fear of being judged or the hope that it will just go away by itself.

One possible and very handy solution for this problem would be a mobile application to help people monitor their activities and how their mood is influenced by certain factors such as people, context, weather and so on. Not only can the application be used by people who think of themselves of being at risk, but also by therapists as an additional tool for their patients.

Studies show that mental health issues are the largest source of global economic burden, greater than cardiovascular and other physical diseases (Mental Health Foundation, 2016) and people that struggle with mental health issues around the

world have limited, if any, chances of accessing psychological help. They are also a public health concern (World Health Organization [WHO], 2014), with a recent study led by the WHO, showing that depression and anxiety disorders alone cost the global economy US$ 1 trillion each year in lost productivity.

One of the most relevant studies in this field is the so-called MeMo study, published on the 30th of July, 2021. This study highlighted the potential utility of apps for clinical practice and the way they can be substitutes for interventional tools, or at a minimum, an adjunct to existing treatments. 23 people with mental health problems and 24 young people without mental health problems participated in this quantitative study.

Participants monitored their mood using a mood-monitoring app twice a day for 3 weeks, which was preceded by a 3-week baseline period. Momentary and retrospective assessments of affect regulation (all participants) and therapeutic involvement were used as outcome measures (patients only).

Patients (n = 7) and their clinicians (n = 6) were interviewed individually after the quantitative study. Thematic analysis was used to examine the interview data. In terms of results, the use of an app of this kind significantly reduced reduced momentary negative mood and retrospectively assessed impulsivity across all 47 participants. Similarly, qualitative feedback suggested that apps could help people with impulsive issues.

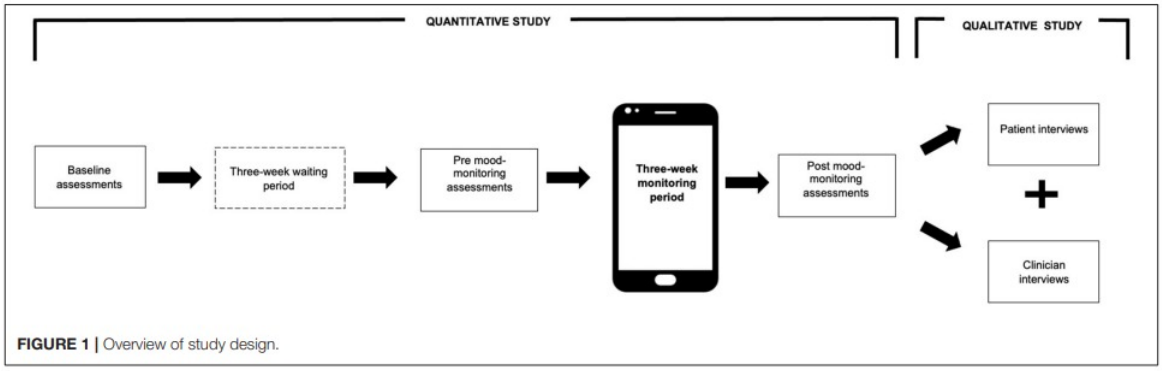


FIGURE 1 | Overview of study design.

2.0.2 Applicability

The usages of this type of application may vary. As mentioned earlier, it might serve as a tool during psychological therapy sessions.

Moreover, it is a great way to encourage people to focus on their mental health too. Most people know they can exercise, sleep well, stay hydrated in order to enhance their physical fitness, but it's less common for people to pay attention to their mental health.

Besides the various methods that are available for solving these kinds of problems, such as meditation, journaling, going out with friends etc , a mobile app wo-

uld the most innovative solution, because people have their smartphones on them 24/7, therefore, it is extremely handy.

2.0.3 Description

This application allows the user to sign up, log in and then start to configure everything everything in order to be able to run an analysis. The app allows the user to add all the people they usually interact with, such as friends, parents, siblings, partner etc. Then, after having a few people in the database, they can plan events , by mentioning all the people that will take part of them and by setting the date and time of each event. When an event ends, the user is obliged to set a grade for it, depending on their mood during the event.

After evaluating a few events, the user will be able to run an analysis and see how certain factors such as time of the day, location, people, weather have influenced their mood. When the user runs an analysis, the data about all the events will be passed through a statistical layer and then the results will be printed in some charts. When the user starts an event, the application will get the location and the time by itself and add all this data the database.

Capitolul 3

Technical Part

3.1 Front-end technologies

3.1.1 Angular

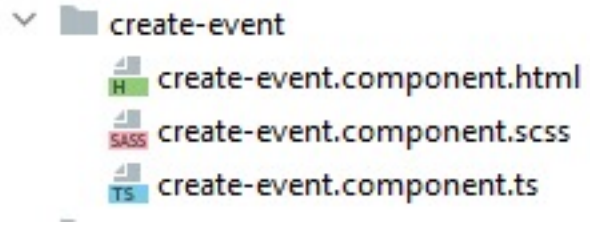
The development of the front-end part of the application was achieved using Angular. Angular is an open-source Javascript front-end framework used to create single page applications. Some of the most popular websites that are built using Angular are g-mail, Upwork or Nike. It is designed for web, desktop and also mobile platforms.

The first and most useful feature that Angular provides is its using the regular DOM, therefore, it updates it every-time a change is made in one of the containing project HTML files.

Typescript would be another reason to choose Angular. It is very useful because it allows the developer to write Javascript code that is easier to understand, maintain or test. It compiles down to Javascript code that can be run on any platform, so it is not mandatory to use, but highly recommended.

Last, but not least, data binding is a very interesting advantage that Angular brings to the table, due to the fact that is used two-way binding. Data binding allows an Internet user to manipulate HTML elements using the browser and it is used for web pages that used dynamic HTML. Two-way data binding is important, because every change made in one of the UI element is reflected into the corresponding model state and vice-versa, allowing the DOM to connect to the model data.

Angular allows you to give your app a nice structure by creating so-called components. Basically, your application is a collection of components that interact with each other in different ways. When generating a new component, Angular will generate 4 files: an HTML file for the template, a Typescript file for the logic, a CSS or .SCSS file for styling and a .spec.ts file for testing. Below is an example of a component structure:



3.1.2 Ionic

Ionic is an open-source UI framework for developing performant, high-quality mobile apps using web technologies, with integrations with popular frameworks such as Angular or React.

It was created in 2013 and it is the world's most popular cross-platform mobile development technology stack, with over 5 million apps being developed using it. Ionic makes it very easy to create mobile apps without having to learn new skills.

It provides a large set of components to use and define the behaviour and user experience of your application. Components vary from small ones, such as button, checkbox, badge to larger ones, like modal, tabs, card etc.

These components can be customized by the developer by setting their properties in HTML, or setting them up using configuration options, depending upon the type of object. Moreover, depending on their nature, they can be populated with custom values.

One other advantage would be that Ionic can build multiple apps from one codebase, which helps with long-term maintenance. If there is a bug, developers have to solve it in only one place instead of many. This way, it is easier to maintain a robust and clean build.

3.2 Back-end technologies

3.2.1 ASP.NET CORE

ASP.NET Core 3 is a cross-plaform, high-performance, open-source framework for building modern, Internet-connected apps. It was first released in 2016 and is a redesign of earlier Windows-only versions of ASP.NET which runs on macOS, Linux and Windows.

ASP.NET Core provides some very useful features to build web APIs and web apps, such as MVC pattern, which allows you to develop more testable APIs or web apps, built-in support for multiple data formats, which allows your server-side app to be used by a large set of clients, like browsers or mobile devices, model binding, which automatically maps data from HTTP requests to action method parameters

or model validation, which automatically performs client-side and server-side validation.

ASP.NET Core Identity Provider

ASP.NET Core Identity provides a framework for managing and storing user accounts in ASP.NET Core apps. Identity is added to your project when Individual User Accounts is selected as the authentication mechanism. It is an API that supports user interface login functionality and managers users, passwords, profile data, roles, claims, tokens, email confirmation, and more.

By default, Identity makes use of an Entity Framework (EF) Core data model. Users can create an account with the login information stored in Identity or they can use an external login provider, such as Facebook, Google, Microsoft Account, and Twitter.

```
[HttpPost]
[AllowAnonymous]
0 references
public async Task<IActionResult> CheckLogin([FromBody] UserModel loginModel)
{
    UserEntity user = await _userManager.FindByEmailAsync(loginModel.Email);
    if (user == null)
        return BadRequest("User does not exist");

    Microsoft.AspNetCore.Identity.SignInResult result =
        await _signInManager.PasswordSignInAsync(user, loginModel.Password, false, false);

    if(result.Succeeded)
    {
        JwtSecurityTokenHandler tokenHandler = new JwtSecurityTokenHandler();
        var key = Encoding.ASCII.GetBytes(_appSettings.Authorization.Secret);
        SecurityTokenDescriptor tokenDescriptor = new SecurityTokenDescriptor {
            Subject = new ClaimsIdentity(new Claim[]
                {
            new Claim(ClaimTypes.Name, user.Id.ToString()),
            new Claim(ClaimTypes.Role, "SuperAdmin")
                }),
            Expires = DateTime.UtcNow.AddDays(1),
            SigningCredentials = new SigningCredentials(new SymmetricSecurityKey(key),
                                        SecurityAlgorithms.HmacSha256Signature)
        };
        return Ok(tokenHandler.WriteToken(tokenHandler.CreateToken(tokenDescriptor)));
    };

    return BadRequest("Incorrect username or password");
}
```

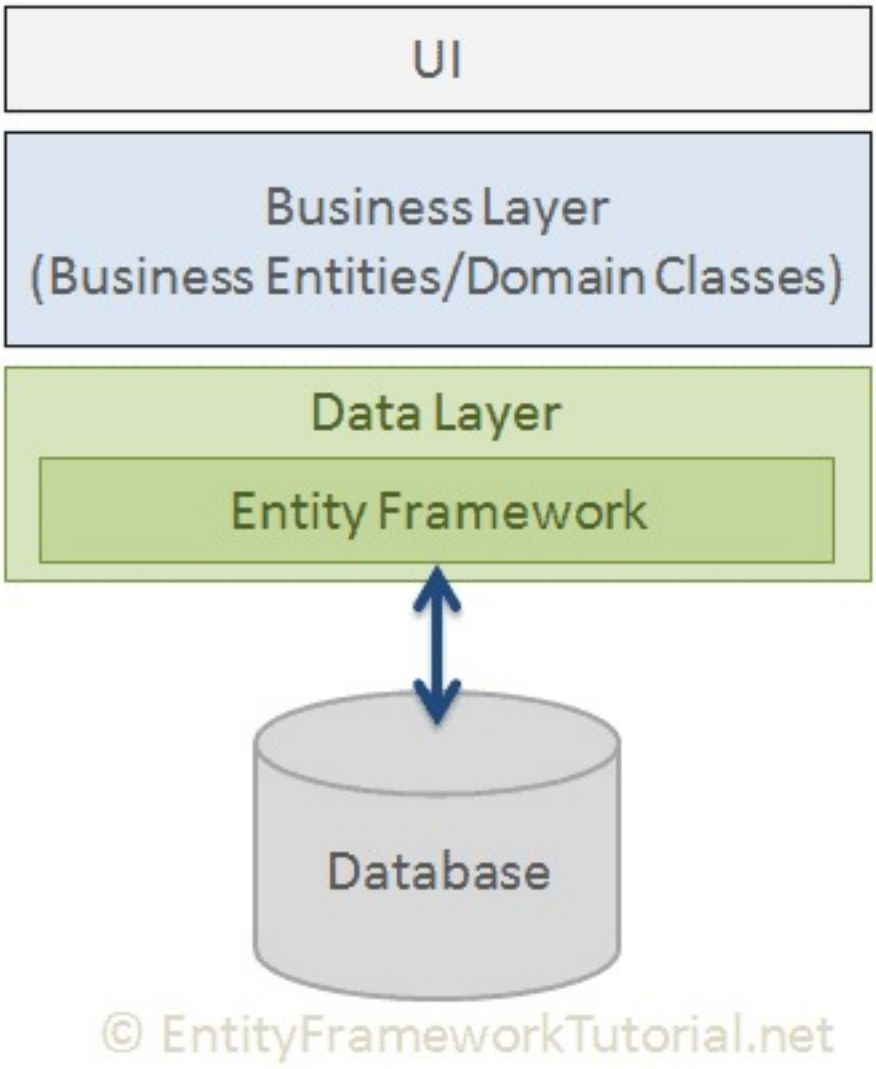
example of endpoint method that uses Identity Provider classes to check if a user is valid

In the image above, some of the usages of Identity Provider classes can be seen. The method ”FindByEmailAsync” of the ”Microsoft.AspNetCore.Identity.UserManager” class is used to get the user that is trying to log into the app. Afterwards, the ”Pas-

swordSignInAsync" method of the "Microsoft.AspNetCore.Identity.SignInManager" class is been used to check if the user has entered the correct password.

Entity Framework Core

Entity Framework Core is a modern object-database mapper for .NET. It allows developers to work with data using objects of domain specific classes without focusing on the underlying database structure. The official definition states that "Entity Framework is an object-relational mapper (O/RM) that enables .NET developers to work with a database using .NET objects. It eliminates the need for most of the data-access code that developers usually need to write." The follwing figure illustrates where the Entity Framework fits into the application:



Some of the most important features are the following:

- LINQ queries:

EF allows developers to use LINQ queries to manipulate data from the underlying database. The db provider will translate these LINQ queries to the database-specific query language. EF also allows us to execute raw SQL queries directly to the database.

- Change tracking:

  EF keeps track of changes occurred to instances of your entities (Property values) which need to be submitted to the database.

- Schema migrations:

  EF provides a set of migration commands that can be executed on the NuGet Package Manager Console or the Command Line Interface to create or manage the underlying database schema.

ASP.NET Core Request Lifecycle

The ASP.NET Core MVC Request Life Cycle is a series of steps that happen every time an HTTP request is handled by the application. The request lifecycle consists of various stages, such as:

- Middleware

  It is software that is used to handle requests and answers in an app pipeline. Each component determines if the request should be forwarded to the next component in the pipeline or it can perform work before and after the next component.

- Routing

  Routing in ASP.NET MVC is nothing more than matching incoming URIs to actions. With the use of convention routes and attribute routes, the routing middleware component determines how an incoming request might be mapped to controller and action methods.

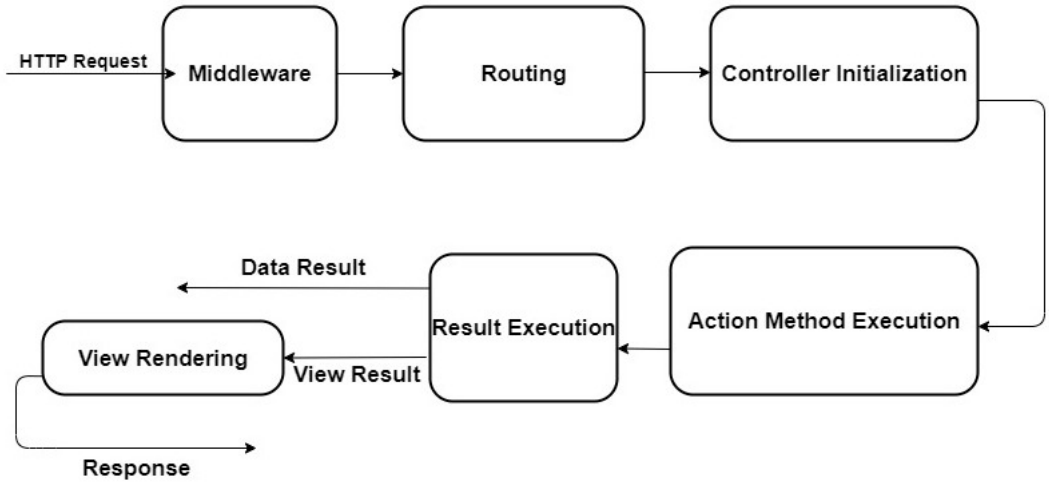
- Controller Initialization

  The process of initialization and execution of controllers takes place at this step of the lifecycle. Controllers are in charge of dealing with incoming requests. On the basis of the route templates provided, the controller determines the relevant action methods.

- Action method execution

  These methos are placed inside controller classesand they execute logic to create and return Action Results.

- Result Execution - During this stage, the response generated to the original HTTP request is executed.



ASP.NET Core MVC Request Life Cycle

3.2.2 Database

postgreSQL

PostgreSQL is an object-relational database management system based on POSTGRES. It was developed at the University of California at Berkeley Computer Science Department and has more than 30 years of active development on the core platform. It supports a large part of the SQL standard and offers many modern features, such as complex queries, foreign keys, transactional integrity and multi-version concurrency control. It also offers the user the possibility to add new data types, functions, operators etc.

3.2.3 HyperText Transfer Protocol (HTTP)

The Hypertext Transfer Protocol (HTTP) is an application-layer protocol used to send hypermedia documents like HTML. It was created to allow web browsers and servers to communicate, but it can also be used for other reasons. HTTP is based on the classic client-server model, in which a client establishes a connection to make a request and then waits for a response. HTTP is a stateless protocol, which means that between two requests, the server does not maintain any data (state).

An http request can be of different types:

- GET - used to retrieve resources; additional data can be passed through the request header

- POST - used to send information to the server for operations of type 'create'. The data is attached after the headers, in a place called 'body'
- PUT - same as POST, but used for update operations
- DELETE - used to send data such as ids for delete operations

An http call also includes a resource address represented by the URL. A URL (Uniform Resource Locator) represents the unique address for a web page, also known as an Internet resource.

It's general form is: protocol://domain-name:port/directory/resource, the protocol is usually HTTP or HTTPS (secure HTTP), the domain name, the port number directory, and resource specify the the exact place where the resource on the destined computer is located.

After processing a client's request the server returns a response. An HTTP response can contain several headers that specify the allowed methods, content length, type and, encoding. In addition, it has a status code.

An HTTP status code is a server response to a browser's request. When you visit a website, your browser sends a request to the site's server, and the server then responds to the browser's request with a three-digit code. The most usual http status codes are:

- 200 OK - returned when everything worked properly.
- 201 Created - The request has been fulfilled, resulting in the creation of a new resource.
- 204 No Content - The server successfully processed the request, and is not returning any content.
- 400 Bad Request - The server cannot or will not process the request due to an apparent client error
- 401 Unauthorized - The response must include a WWW-Authenticate header field containing a challenge applicable to the requested resource
- 404 Not Found - The requested resource could not be found but may be available in the future.
- 500 Internal Server Error - A generic error message, given when an unexpected condition was encountered and no more specific message is suitable

Capitolul 4

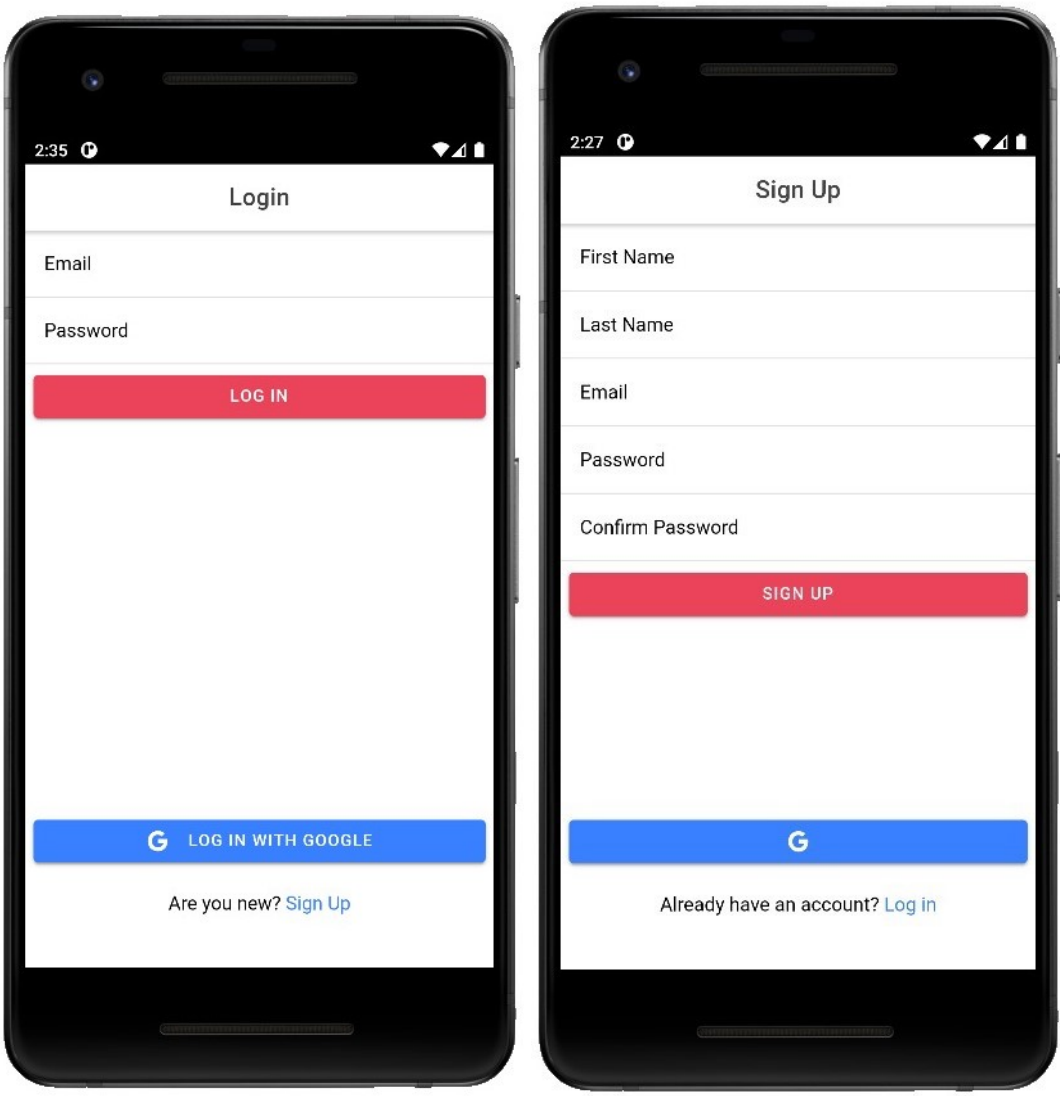
Architecure of the app

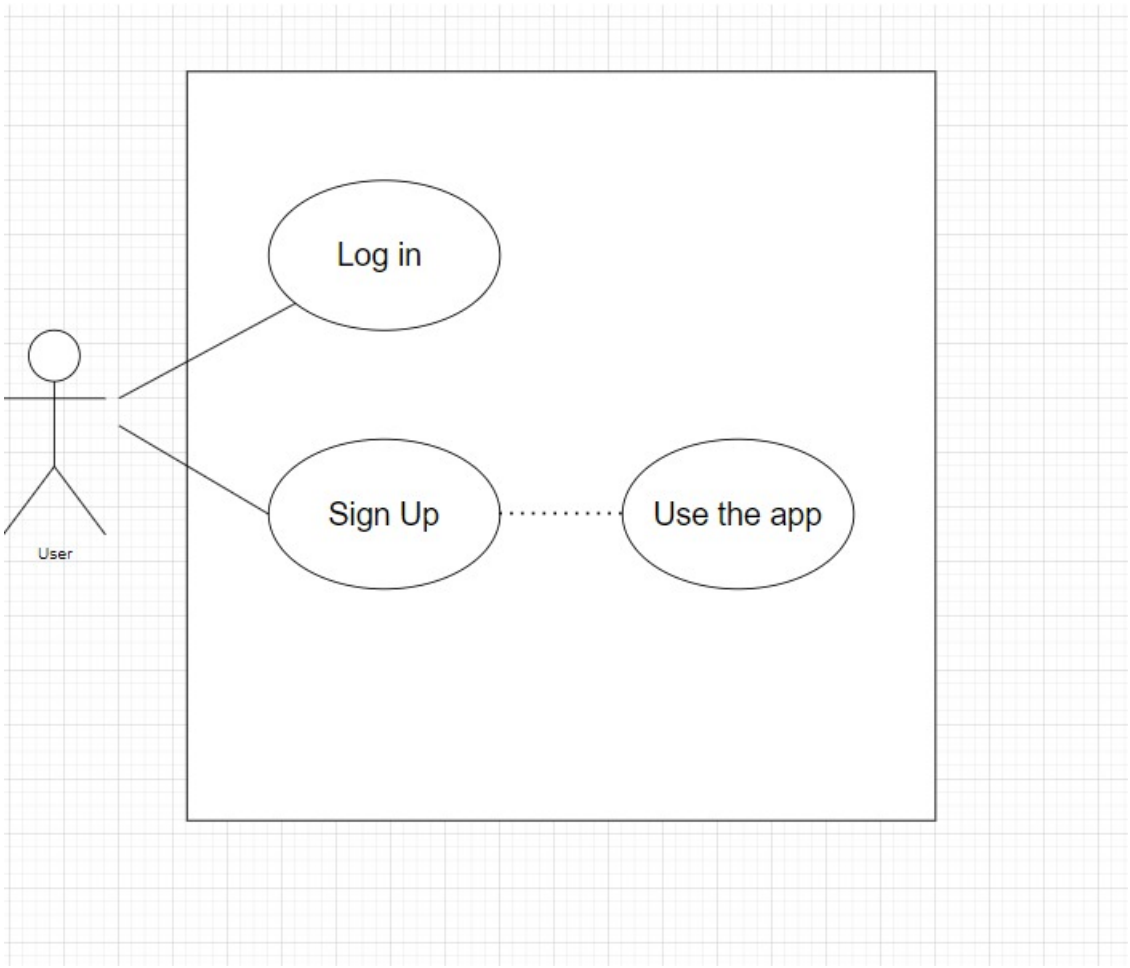
This application uses a simple client-server architecture that communicates through the Hypertext Transfer Protocol (HTTP).

4.0.1 Features

Authentication

When the user enters the app, the login page will be rendered. They can either fill the input fields if they have an account, or they can navigate to the Sign Up page to create an account.


2:35
Login
Email
Password
LOG IN
LOG IN WITH GOOGLE
Are you new? Sign Up
2:27
Sign Up
First Name
Last Name
Email
Password
Confirm Password
SIGN UP
Already have an account? Log in



Authentication use case diagram

4.0.2 Login

After the user types in their username and password and taps the login button, the frontend side of the app will retrieve this data and send it to the server using an HTTP POST request to the ”CheckLogin” endpoint method, which is located in the 'Account Controller'.

All the methods in this controller have the 'AllowAnonymous' attribute, in order to allow access to http requests that don't contain authorization token. In other words, these endpoints can be accessed by any users, regardless of their authentication status.

This endpoint will use the 'FindByEmailAsync' method of the UserManager class to figure out if there actually is a user with that email in the database. If so, it will then use the 'PasswordSignInAsync' method to check if the password corresponds to that certain e-mail and if this function return true, it will create an encrypted bearer token, containing the user's data.

In case one of these 2 operations fails, a 400 bad request code with the corresponding error message will be returned.

On the client side, after a successful login on the server, the authentication token will be stored in the session storage, in order to be accessible throughout the entire navigation of the user in the app. This token will be used in all of the api calls the client makes to the server (except the login or signup ones) for data, because those require a user to be logged into the app and it will be used on the server part to retrieve user-specific data.

For example, a piece of code that creates an HTTP GET request to the GetPeople endpoint looks like this:

```
12
13    public getPeople(): Promise<Array<PersonModel>> {
14      return this.http.get<Array<PersonModel>>( url: environment.api + '/People/Get', options: {
15        headers: new HttpHeaders( headers: {
16          "Authorization": `Bearer ${sessionStorage.getItem( key: 'bearerToken')}`
17        })
18      }).toPromise();
19    }
20
```

On the back-end side, an endpoint that does not handle login or signup operations looks like in the figure below:

```
12  namespace backend.Controllers
13  {
14      [ApiController]
15      [Route("moodapp/api/[controller]/[action]")]
16      [Authorize]
        1 reference
17      public class PeopleController : ControllerBase
18      {
19          private PeopleWorker _peopleWorker;
20
            0 references
21          public PeopleController(PeopleWorker peopleWorker)
22          {
23              _peopleWorker = peopleWorker;
24          }
25
26          [HttpGet]
            0 references
27          public async Task<List<PersonModel>> Get()
28          {
29              var userId = User.FindFirstValue(ClaimTypes.Name);
30              return _peopleWorker.GetPeople(userId);
31          }
32
33          [HttpPost]
            0 references
34          public void Add([FromBody] PersonModel personModel)
35          {
36              var userId = User.FindFirstValue(ClaimTypes.Name);
37              _peopleWorker.AddPerson(personModel, userId);
38          }
39
40          [HttpPost]
            0 references
41          public void Update([FromBody] PersonModel personModel)
42          {
43              _peopleWorker.UpdatePerson(personModel);
44          }
45
46          [HttpDelete]
            0 references
47          public void Delete([FromQuery] Guid id)
48          {
49              _peopleWorker.DeletePerson(id);
50          }
51      }
52  }
```

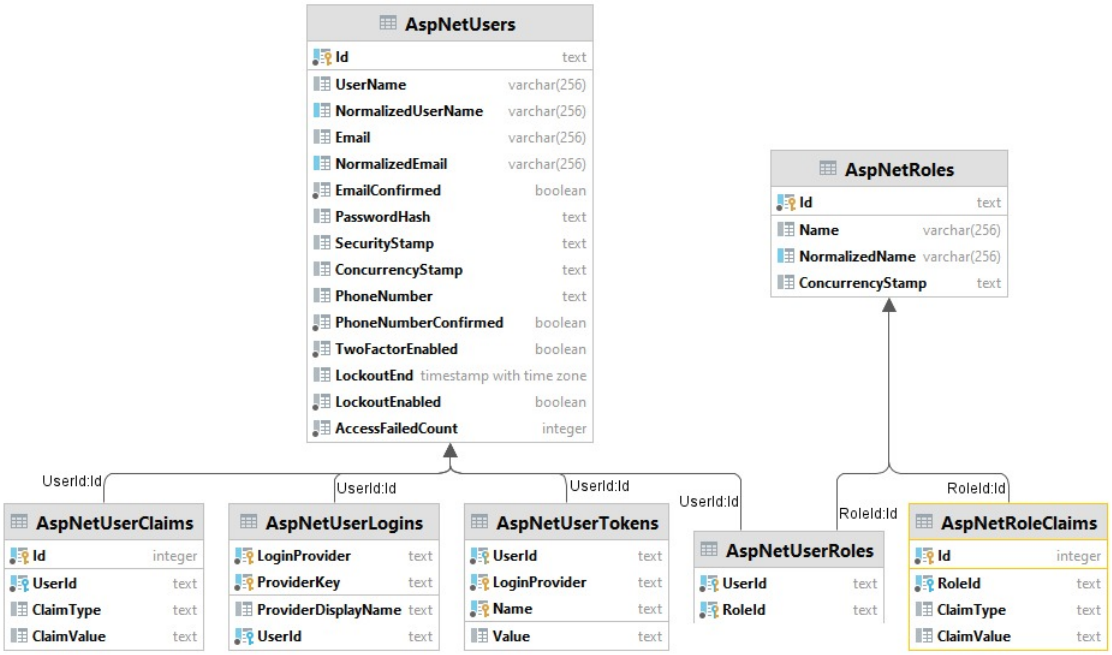
As visible above, the controller class contains the 'Authorize' attribute, which means that no unauthorized http call is allowed. Any call to this endpoint that does not have a bearer token set, will get a 401 Unauthorized status code.

4.0.3 User Management

Frontend-wise, the application allows the user to perform sign-up and log-in. After a successful sign-up, the user will be redirected to the login page, in order to fill in the previously created e-mail and password.

Backend-wise, the .NET Core Identity Provider provides built-in authentication and authorization features. These are all configured in the 'Startup.cs' file and they are dependent on a few NuGet Packages, such as 'Microsoft.AspNetCore.Identity.EntityFramewor or 'Microsoft.AspNetCore.Identity'.

After writing the sql script to create the tables that have no connection with users, the generation of all the models and the database context class will take place. Then, we pass our database through a so-called 'Identity Migration', which will generate all of the necessary database tables that the application needs in order to perform authentication operations.

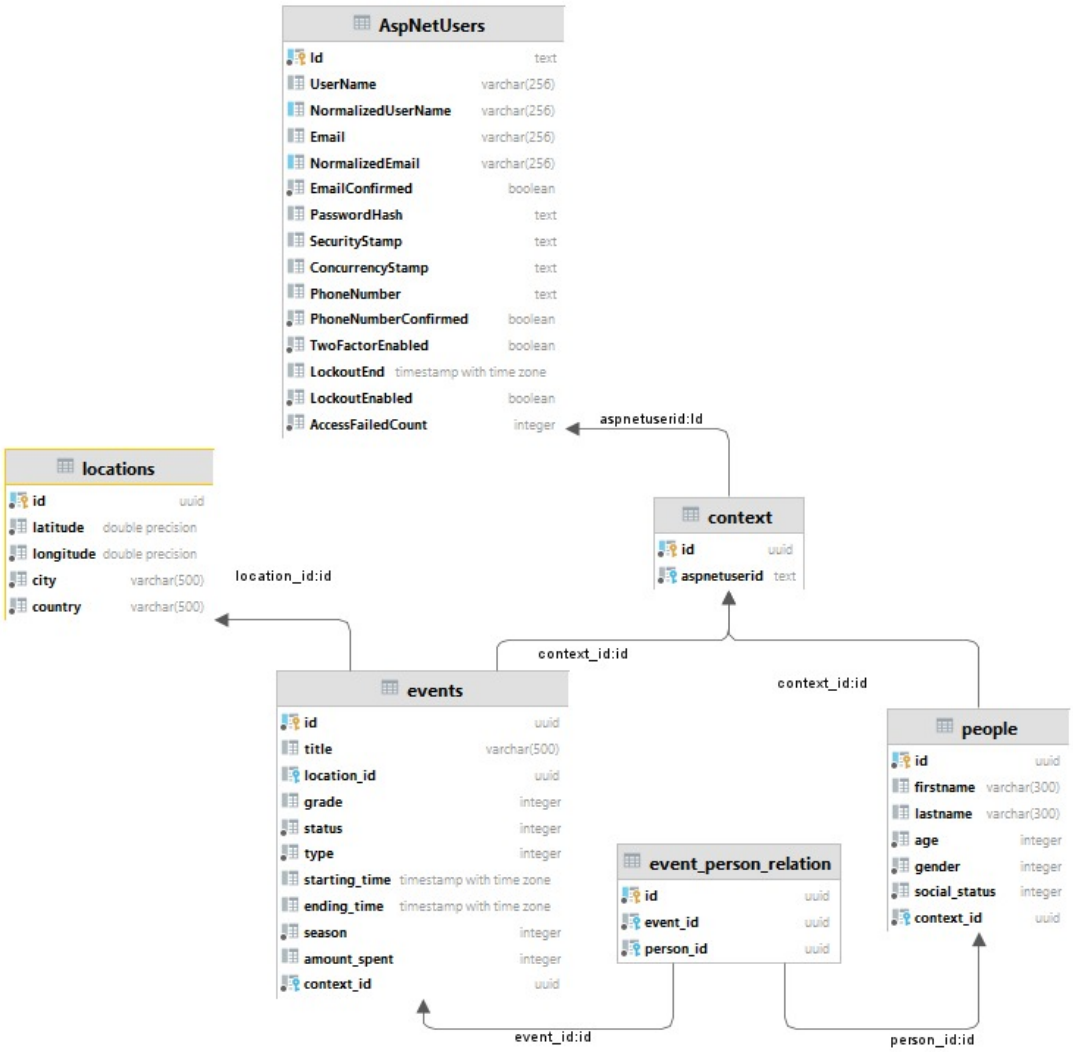


db schema of the auto-generated identity tables

From then on, all operations that request resources from the db will take place in the context of a user. As visible in the figure below, the app makes use of the so-called Context database table to make the retrieval of Person or Event records easier.

Every time a client requires such records from the server, the server will get the user's id from the Authorization (Bearer) token sent on the request, and it will use it to filter all the data to be returned. The same thing happens in case of operations that require insertion into the database, because, as seen below, the tables Event and Person have a column called context id, which refers to the Context table created when the user has signed up in the application.

Only Update or Delete operations do not need the user id, because events and people have ids, and nothing except that is needed in order to perform such operations.



Database structure

4.0.4 Domain

Capitolul 5

Conclusions

Technology is a huge advantage of today's era and people might as well use it as solutions to problems that have been around forever. Given the widespread usage of smartphones and tablet devices among the general population, mHealth has a huge potential to be an effective tool in reducing issues such as anxiety, stress, alcohol disorders, sleep disorders, depression or PTSD (post-traumatic stress disorder). With the increasing deficiency in mental health professionals globally, a mobile phone-based application that could monitor mental health will go a long way in reducing the rate of mental illness among hard-to-reach population.

Bibliografie

[1] Wang K, Varma DS, Prosperi M, A systematic review of the effectiveness of mobile apps for monitoring and management of mental health symptoms or disorders, Journal of Psychiatric Research (2018), doi: https://doi.org/10.1016/j.jpsychires.2018.10.006.

[2]

[3]

[4]

[5]

[6] Leslie Lamport (1994) *LaTeX: a document preparation system*, Addison Wesley, Massachusetts, 2nd ed.