Java – Object Oriented Programming

Smart E-health Consulting System



**Lecturer:** Luigi La Blunda

**Team 25:**

* Htet Htet Win: 1177149
* Dwipa Flügel: 1322033
* Abdul-Rahman Oudeh: 1340277
* Nihal Ahmed Alizada 1352320



Structure

[1. Project Description 3](#_Toc95754956)

[2. Project Motivation 3](#_Toc95754957)

[3. Requirements 3](#_Toc95754958)

[4. Team Organization 4](#_Toc95754959)

[5. E-Health Application Overview 4](#_Toc95754960)

[5.1 Welcome Page: 5](#_Toc95754961)

[5.2 Registration Page: 6](#_Toc95754962)

[5.3 Login Page: 6](#_Toc95754963)

[5.4 Login as Admin: 7](#_Toc95754964)

[5.5 Access/Edit/Delete User Profiles: 7](#_Toc95754965)

[5.6 Login as Patient: 8](#_Toc95754966)

[5.7 Add/Shift/Delete Appointment: 9](#_Toc95754967)

[5.7.1 Adding an Appointment: 10](#_Toc95754968)

[5.7.2 Shifting an Appointment: 10](#_Toc95754969)

[5.7.3 Deleting an Appointment: 11](#_Toc95754970)

[5.8 Save Appointment as PDF File 11](#_Toc95754971)

[5.9 Add Health Information 12](#_Toc95754972)

[5.10 Save Health Information as PDF File 12](#_Toc95754973)

[5.11 Emails 12](#_Toc95754974)

[6. Technical description of solutions: 13](#_Toc95754975)

[6.1 Geocoding: 13](#_Toc95754976)

[6.2 Nearby Search: 13](#_Toc95754977)

[6.3 JSON Parsing: 14](#_Toc95754978)

[6.4 Hashing: 14](#_Toc95754979)

[6.5 PDF File 15](#_Toc95754980)

[7. Conclusion 17](#_Toc95754981)

[8 Libraries 17](#_Toc95754982)

[9 References: 17](#_Toc95754983)



**List of Figures:** 

[Figure 1 Activity Diagram 5](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756419)

[Figure 2 Welcome Page 5](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756420)

[Figure 3 Error Message 6](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756421)

[Figure 4 Registration Page 6](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756422)

[Figure 5 Login Page 6](#_Toc95756423)

[Figure 6 Login as Admin 7](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756424)

[Figure 7 Access User File 7](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756425)

[Figure 8 Delete User Files 8](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756426)

[Figure 9 Edit User Files 8](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756427)

[Figure 10 Appointment Management 8](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756428)

[Figure 11 Login as Patient 8](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756429)

[Figure 12 Select Health problems and Search Radius 9](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756430)

[Figure 13 Select Doctor 9](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756431)

[Figure 14 Add Date, Time, and Reminder 10](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756432)

[Figure 15 Pop up after adding Appointment 10](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756433)

[Figure 16 Shift Appointment 11](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756434)

[Figure 17 Delete Appointment 11](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756435)

[Figure 18 PDF File of Appointment 11](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756436)

[Figure 19 Add Health information 12](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756437)

[Figure 20 Confirmation Mail 12](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756438)

[Figure 21 Geolocation 13](#_Toc95756439)

[Figure 22 Nearby Search 14](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756440)

[Figure 23 Excerpt from code: encryptX Function 15](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756441)

[Figure 24 Excerpt from the code: second method that retrieve data from Database 15](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756442)

[Figure 25 Excerpt from code: Create a PDF 16](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756443)

[Figure 26 Libraries 17](https://d.docs.live.net/b52fde823f83033e/Dokumente/Java_Doc.docx#_Toc95756444)

**List of Tables:**

[Table 1 Tasks distribution 4](#_Toc95747333)

## 1. Project Description

Appointment scheduling with your preferred doctor has never been so easy with the E-health Appointment System. The Smart E-health Consulting system has all the functionalities that a small doctor’s office or clinic needs. The patient can schedule an appointment of their choice with their preferred doctor in just a few clicks. E-health is user-friendly and provides important functions such as appointment scheduling or searching for doctors or specialists based on health problems within a given search radius and many more exciting features. With our E-health appointment system, doctor offices and clinics can save time and headaches as the patients can schedule their own appointments online and update their information immediately while being at home. No more waiting in person or calling just to create or cancel an appointment. A perfect fit for the current pandemic crisis.

## 2. Project Motivation

Predicting what the future holds for technology is difficult. Especially at times like this where the pandemic crisis has changed many aspects of our life. People depend more and more on technology. We as a team want to be a part of the world of technology, therefore we have been thinking of creating a program which could make people's daily life simpler. We came up with the idea to create the Smart E-Health Consulting system.

## 3. Requirements

Before building a program and starting the work on the code it is important to define the requirements that are needed for the program we are going to develop. As we started our journey, we began by making a list of basic requirements that we wanted in our program as well as those requirements that cover project requirements. First, we wanted our program to have the ability to let users register an account, along with logging into these user-registered accounts as well as predetermined admin-accounts. It is also necessary that the login data for both admin and user need to be securely encrypted as well. Next the user should be able to make an appointment for a specialized doctor based on the medical issues they are having. Additionally depending on the given area of search only the doctors within the search radius of the registered address should be displayed for them to make an appointment. Afterwards the ability to shift or cancel an appointment should also be included. For the admin it must have the capability to manage (access, edit, delete) all the user profiles. Finally, all the health information and appointment data can be exported as a PDF file. Overall, our goal is to have a program that is easy to use and at the same time extremely beneficial to the users. Not only the patients but also the admins should have a clear overview of the application. By collecting the requirements that were needed as the very first step, we thought it will be advantageous to us in the future process of creating the program. 

## 4. Team Organization

As a team we all have our assigned parts, and we met at least once a week to show our progress and accomplishments to the group while expressing our opinions and making suggestions. We also have a WhatsApp group chat where we exchanged each other about the project. If a problem occurred, we would solve it all together. Multiple members worked on some parts of the code and documentation. For the documentation each of us wrote a section about their assignment. After we finished writing, we proof-read our documentation as well. In the following table you will see the distribution of our parts.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tasks for Project and Documentation** | **Win** | **Oudeh** | **Alizada** | **Flügel** |
| Welcome and registration page |  | x | x |  |
| Login page for patient |  |  | x |  |
| Login page for admin |  | x |  |  |
| Access, edit and delete option for admin |  | x |  |  |
| Add health problems and select doctor as patient |  |  |  | x |
| Add appointment date, time, and reminder | x |  |  |  |
| Shift and cancel option for appointments | x |  |  |  |
| Emails for confirmation, changes, and reminder | x |  |  |  |
| Add health information |  |  | x |  |
| PDF File for health information and appointment | x |  |  |  |
| Geolocation, Nearby Search, JSON Parsing |  |  |  | x |
| Encryption for password |  |  | x |  |
| Project Description and Project Motivation |  | x | x |  |
| Requirements and Conclusion | x |  |  |  |
| E-Health Application Overview and activity Diagram | x | x | x | x |
| Technical Description | x | x | x | x |

Table 1 Tasks distribution

## 5. E-Health Application Overview

Before we go further into the technical descriptions, we would like to introduce how our program overall works through the activity diagram and show afterwards how every section of our program looks as we explain it step by step.

## 5.1 Welcome Page:

Figure 1 Activity Diagram

Upon running the program, the user comes across the welcome page first. He has the option to register an account when he does not have one or login directly when he already created an account. 

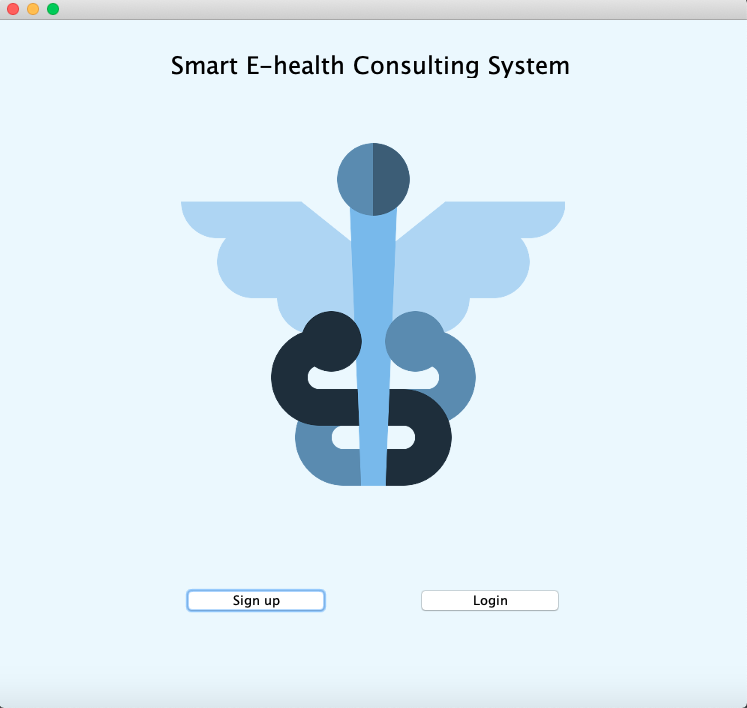
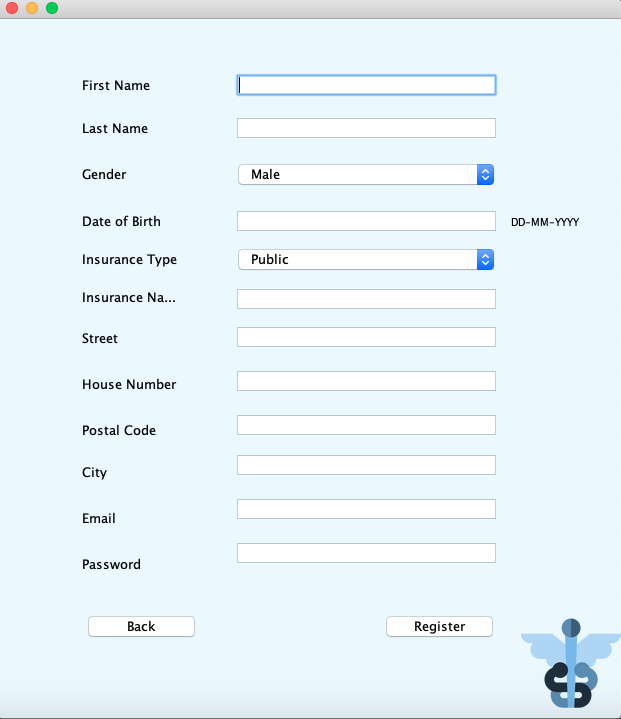


Figure 2 Welcome Page

## 5.2 Registration Page:

****

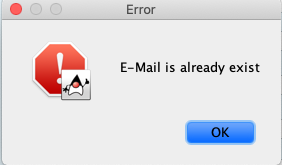
The patient can create an account by entering their first and last name, date of birth, gender, insurance name and type, their address which includes street, house number, city, and postal code. They must enter a valid email address and a password. The patient can't register with an already existing email. All this information is stored in our database and the user can use our services after successful creation of an account. 

Figure 3 Error Message

Figure 4 Registration Page

## 5.3 Login Page:

There are two options on the login page. The user can either login as admin or as patient.Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 5 Login Page

## 5.4 Login as Admin:

The admin cannot create an account, however their credentials are already stored in the database and upon logging in, they are compared to the one in the database. After logging in they will be forwarded to the “User Management Page”.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 6 Login as Admin

## 5.5 Access/Edit/Delete User Profiles:

Ein Bild, das Tisch enthält.

Automatisch generierte BeschreibungAfter the admin has successfully logged in, they have 3 options to choose. They can access all the patient profiles, seeing the patient's general information like their first name, last name, gender, date of birth, insurance name and email. They can also edit their profiles. For that, they must enter the email of the patient to edit the profile for example first name, last name, and address. The admin can also delete patient profiles. All they have to do is to enter the patient's first name, last name, and email.

Figure 7 Access User File

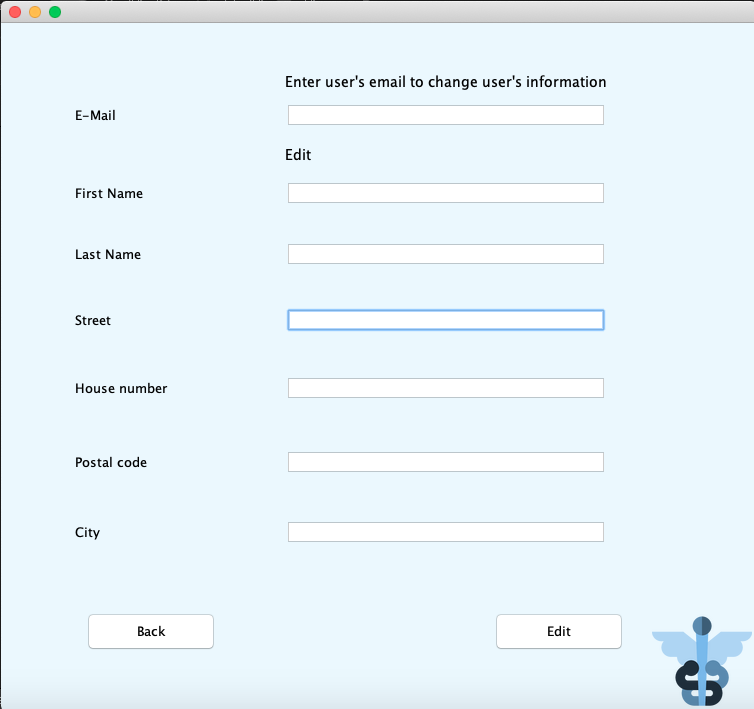
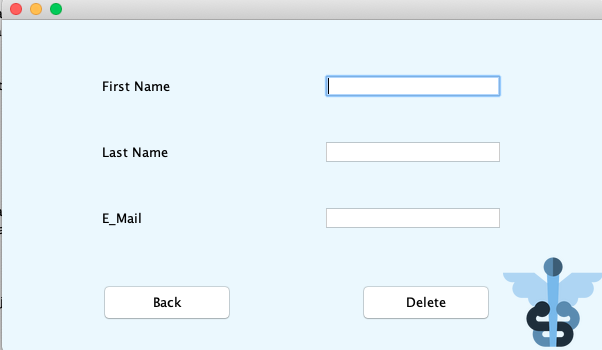


Figure 8 Delete User Files

Figure 9 Edit User Files

## 5.6 Login as Patient:

Ein Bild, das Text enthält.

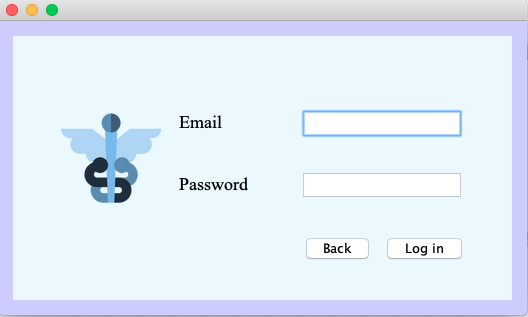
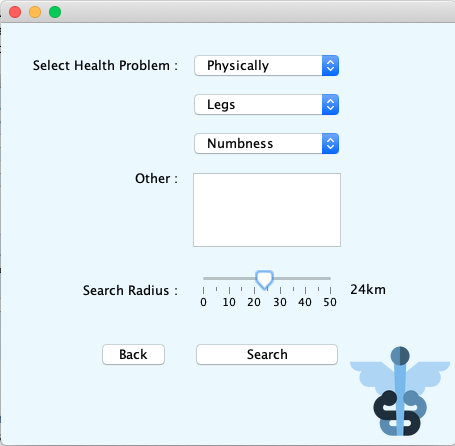
Automatisch generierte BeschreibungThe patient logs in with their email and password. After logging in successfully, the patient is redirected to the main dashboard which consists of 4 possible options, which are adding health information and making, editing, or deleting an appointment. 

Figure 10 Appointment Management

Figure 11 Login as Patient

## 5.7 Add/Shift/Delete Appointment:

After the patient has successfully logged in, they can decide whether they want to add an appointment, shift, or cancel an already made appointment. If the patient decides to add an appointment the patient will be directed to a page where they can choose to specify their search for an appropriate specialist.

In this new window the user is able to choose a very broad or very specific pre-determined health problem or if not available, is able to input their health problem in a text field, which is later used as a keyword in the search for appropriate specialists, as well as being able to select a search radius by using a slider.

When the user has finished specifying their current problem by pressing the “Search” button, they will be redirected to a new window showing them a list of specialists (if any are available) in accordance with their search criteria.

Figure 12 Select Health problems and Search Radius

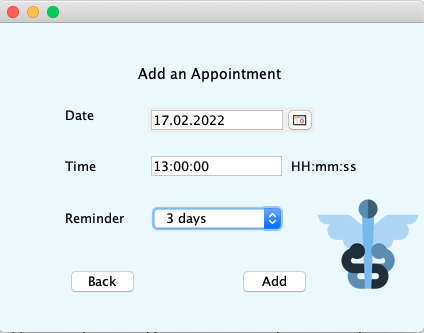
From this point on, they are able to select a single specialist out of the provided list and choose them as their desired doctor and they will be redirected to the “Choose a date” page to finish the appointment-making process.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 13 Select Doctor

## 5.7.1 Adding an Appointment:



In this page the patient can select the desired appointment date, fill the time in and select the reminder, which is displayed with the drop-down menu with all the selections (1 week, 3 days, 1 hour or 10 minutes).

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungAfter adding an appointment, the window that says “You have successfully added an Appointment. Your confirmation mail will be sent to your email shortly” will be displayed and the added appointment will be stored in the corresponding database. There they will also find the button where they can also save the appointment data as a PDF file. If the patient decides not to add an appointment after all they can also go back to the main dashboard.

Figure 14 Add Date, Time, and Reminder

For the appointment date and time, the patient can only select the date and put the time in that is either today or in the future. If the patient chooses a reminder time that is not possible, for example the patient wants to be reminded a week before an appointment, but the appointment will take place next day then the patient might need to change the reminder option they selected or else they will get an error message hinting the mistake and won’t be able to make an appointment.

Figure 15 Pop up after adding Appointment

## 5.7.2 Shifting an Appointment:

Next the patient can shift the appointment they have already made to another date or time. On the “Shift the Appointment Page” they can choose the appointment through the drop-down menu where all the appointments they have arranged will be shown and after choosing the appointment they want to shift, the patient can fill in the desired date, time, and reminder of the new appointment. Afterwards the old appointment data will be replaced with new appointment information. It is the similar situation where the patient can only shift an appointment to today or to the future date, time, and appropriate reminder time otherwise the patient can’t shift the appointment.  

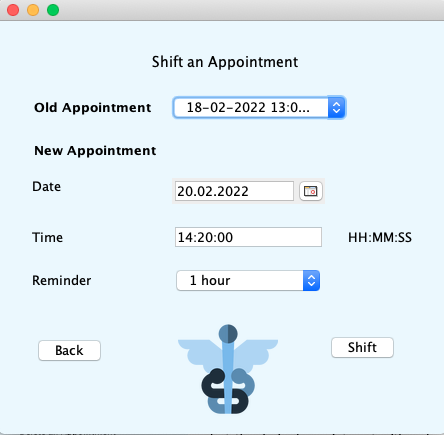


Figure 16 Shift Appointment

## 5.7.3 Deleting an Appointment:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Lastly the patient can delete the appointment on the “Delete an appointment Page”, where they are again given the choice to select the desired appointment with a drop-down menu and can then cancel them after selecting. After confirming the cancellation, the appointment will be deleted from the database and the patient will shortly receive an email for the canceling. 

Figure 17 Delete Appointment

## 5.8 Save Appointment as PDF File

After the patient has successfully made a doctor appointment, the patient can have their appointment data saved as a PDF File. On this file the name and address of the doctor as well as the date and time of appointment are included. The PDF File will then be saved with the last name of the patient and the date of the appointment, for example: “Appointment\_Win\_13-02-2022.pdf”, so the patient can easily find the file they just created.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 18 PDF File of Appointment

## 5.9 Add Health Information

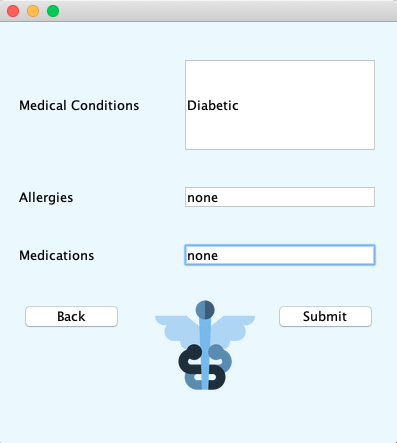
The patient can add any health information regarding his medical conditions, allergies, and medications. Adding any health information is of course optional. The data will be then stored in the database. 

Figure Add Health information

## 5.10 Save Health Information as PDF File

After adding the health Information, the patient can additionally save that information as a PDF File by clicking the Button “Save as PDF”. If there is no health information stored in the database, the patient can’t save it as a PDF file unless they added them. 

## 5.11 Emails

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungAfter the patient has made an appointment, they will receive a confirmation email. After the patient has shifted or deleted the appointment, it is important for the patient to receive an email acknowledging those changes as well. Since the reminder time has been set while making an appointment the patient will receive a reminder email according to the time they have selected. 

Figure 20 Confirmation Mail

## 6. Technical description of solutions:

Since we have presented our E-health application step by step we wanted to furthermore describe our challenges as well our approach to solve them. 

## 6.1 Geocoding:

There’s a variety of Geocoding Libraries online, which is the process of converting a given address into a geographical address using latitude and longitude. In our project we decided to use the Google Maps API, because it provides a variety of functions which were exactly what we needed in our case, and we can avoid using a variety of different libraries. For this we created a class called *Geocoding*, whose methods use the functionalities from the Google Maps library. In order to get the geolocation of any given address we created a method called geolocation, which uses the String parameters Street, House number, Postal Code and City that then creates a string for the appropriate URL that we’re using in our request to the google servers. We then receive a JSON response that we will have to parse which will be explained down further and save the needed latitude and longitude from our address into 2 different float variables, which we then parse into a string in the appropriate format for further requests.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 21 Geolocation

6.2 Nearby Search:The Google Places API provides a variety of search functions for places in their database such as the “Nearby Search” which searches matching places in accordance with the parameters of the URL request that we provide. The problem occurred when we realized that the response not only didn’t send out the formatted address that we needed but a huge amount of unnecessary information as well. So, we picked out the unique Place-ID that an address has from the Nearby Search and made a “Place Details” request using it in order to request the name and the formatted address which we then used to create a *Doctor* instance in an array we created earlier. We further proceed by making a Place Details request for every single result we got from our Nearby Search using a for-loop and creating new instances of doctors in our array in order to create a list of doctors with their names, addresses and phone numbers that Ein Bild, das Text enthält.

Automatisch generierte Beschreibungwe then return as the result of our method. [1] 

Figure 22 Nearby Search

## 6.3 JSON Parsing:

We initially thought of using GSON because we wanted to keep using Google provided Libraries, but after a bit of research we found out that GSON is a Library that is very efficient at deserializing JSON Objects into Java Objects using its fromJson method. The problem that occurred was that we needed to create a class for every JSON Object and JSON Array, imitating its structure and we would have had to make instances for every attribute of our Nearby Search response that represents another JSON Object or JSON Array, which would have been a huge amount of resources just to model our JSON response, especially considering the amount of unwanted data contained in the response.

So after deciding to abandon GSON we searched for an easy-to-use library which would have allowed us to only look for our desired data without much effort and that's when we found the org.json library. Creating a JSON Object using its constructor and using a formatted string of our JSON response as a parameter, it can easily model a JSON Object. It also provides helpful methods like getJSONArray and getJSONObject that create a new instance of those attributes containing our data which we used to specifically search for what was needed, like the latitude and longitude of our user registered address or the name and address of the user selected doctor.

## 6.4 Hashing:

We had a variety of algorithms to use for password encryption. We first used MD5 hashing algorithm but upon reading more about it, it has been officially announced insecure and has been breached by hackers in the past therefore we decided to go with SHA-512 algorithm which is considered a more secure password encryption method.

The first step is to use MessageDigest which is in the *java.security.MessageDigest library* with the getInstance method along with the algorithm that we are using, SHA-512 algorithm in this case. After that is the messageDigest method called to calculate messageDigest on the input string or password. The string bits are then converted to bytes and stored into an array of string.

The byte array is then converted to signum representation using BigIntegar, for which we will import the *java.math.BigInteger* library. The messageDigest is then converted to hex value and 0’s is added to make it 32-Bit. The 32-Bit encrypted password is stored in our database. Exception is thrown if the required Algorithm is not available in the library. [2]

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Figure 23 Excerpt from code: encryptX Function

##### 

## 6.5 PDF File

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungHere we want to describe the class PDF. To be able to create a PDF file firstly we need to import a java library *iText*. There’s a variety of libraries available to use but we decided to use the iText library, which is one of the most popular libraries to use for PDF file creation. To create a PDF file, we used two methods. The first method we use is a method that connects to the database. In the next method we invoke the first method to get the relevant data from the patient, which in this case is either the appointment together with the name and address of the doctor or the health information of the patient depending on the PDF file the patient wants to create.

Figure 24 Excerpt from the code: second method that retrieve data from Database

Next, we are using the *Document* class to create a document as we are setting for the first argument the page size A4 and the next arguments left, right, top, and bottom margins as 50. The string *filename*, which is the name of the document that will be saved on the computer of the patient, is initialized. Afterward *PdfWriter.getInstance()* creates a document with the given filename. Before we add any content to the document, we need to open a Document. After that to add the data we have invoked from the database, we are using the method called *add(new Paragraph(.....)).* Here we are adding the contents of the file.

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungLastly the document is closed using the *close()* method and with that nothing can be written on the file anymore.[3] When we are creating a PDF File with health information instead of the appointment, we also first need to get the allergies, medications and illness from the patient which are saved in the database. Other parts of the code stays the same to create the PDF file.  

Figure 25 Excerpt from code: Create a PDF

After some research on the internet, we started coding for the PDF class. We had no problems with creating the PDF file itself but rather we had some problems retrieving the correct appointment from the database as some of our team members couldn't get the right one invoked even though we used the same code. We tried numerous codes, but it didn't work. In the end we decided to assign a number to each appointment in the database table so we can get the last appointment the patient created by arranging the appointments in descending order.

## 7. Conclusion

We could learn a lot from doing this Java project. At first it wasn't easy for us because none of us have any experience doing a program from scratch but since we started our project by collecting the requirements it shortened us many times and we all had a clear idea of what we wanted. To be able to really start with the project everyone made their own research online and shared the idea as we met at least once weekly. Even though we had our own part we still helped each other out be it with coding or any problems that surfaced along the way. It was important for us too, to learn something together and not just for the grades. Of course, there were some challenges as well, but we could overcome them together as a team. With this project we could deepen our knowledge on coding with Java language and get to know how to develop a program with NetBeans IDE. In the end every one of us had fun coding and designing for this java project and even wished we had more time so we can further improve and make it even more user friendly. All in one we are grateful we earned much fundamental knowledge that could become more practical in future.

## 8 Libraries

We used the following libraries:

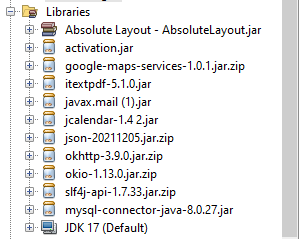




Figure 26 Libraries

## 9 References:

Logo: https://www.pngrepo.com/svg/102720/pharmacy

[1] Author: Google Last Accessed: 02.02.2022

https://developers.google.com/maps/documentation/places/web-service/search-nearby 

[2] Author: RishabhPrabhu Last Accessed: 25.1.2022 http://www.geeksforgeeks.org/sha-512-hash-in-java

[3] Author: Sonoo Jaiswal Last Accessed: 08.02.2022 https://www.javatpoint.com/java-create-pdf

**MySql connector:** https://dev.mysql.com/doc/connector-j/8.0/en/connector-j-overview.html

**JavaMail:** https://javaee.github.io/javamail/#API\_Documentation

**JCalendar:** https://toedter.com/jcalendar/

**ITextPDF**: http://www.java2s.com/Code/Jar/i/Downloaditextpdf510jar.htm

**Activation:** https://www.oracle.com/java/technologies/java-beans-activation.html#download

**JSON:** https://jar-download.com/artifacts/org.json

**Okhttp:** https://jar-download.com/artifacts/com.squareup.okhttp3/okhttp/3.9.0/source-code

**Okio:** http://www.java2s.com/ref/jar/download-okio1130jar-file.html

**Slf4j:** http://www.java2s.com/Code/Jar/s/Downloadslf4jjar.html

**Google maps:** https://github.com/googlemaps/google-maps-services-java