3nd Assignment JavaBasicsIII Documentation

Project Participants:

- **Κριστιάν Γκολέμι Π18029,**
- **Χρήστος Μιχαήλ Καταγής Π18067,**
- **Μιχαήλ Κατσούλας Π18071,**
- **Αλέξανδρος Γκινέτσι Π18028**

Project Description:

This project is an extension of the JavaBasics and JavaBasicsII projects. It is a Dynamic Web Project which means that it is built upon technologies like Java Servlets, JSP files, filters and associated metadata, in addition to static resources like HTML and CSS files. This project tries to simulate an online platform which is accessed by patients, doctors and administration users in general. Using this platform, all users can register by providing the appropriate information, so later on they can login using a username and a password to do certain tasks regarding their role. For example, patients can login to book appointments with the available doctors, while also having the ability to manage these appointments. Doctors make their availability known so patients can book an appointment with them, they can manage their appointments like patients do too. Finally the platform is also used by admins which are responsible of managing the members of the platform, having the ability to add and delete patients and doctors from the platform.

Code Analysis:

- **Web App Libraries:**

  jsp-api.jar

  mysql-connector-java-8.0.25.jar

  servlet-api.jar

- **Packages:**

  Basic: This package contains the classes Appointment, Doctor, Patient and Users which were showcased and used at the 1st Assignment. The classes Admin and Encryption have been also added. The Encryption class is used to hash a password using the MD5 hashing algorithm. The Admin class sets up the values and the methods used by the Admin objects.

Database: This package contains the Dao.java class. Dao are initials that stand for database access object. This means that the role of this class is being responsible of accessing and connecting to the database in order to insert a new user/patient or validate the credentials of a user/patient. This class consists of 7 functions: 3 insert(Class obj) functions (insert(Patient patient), insert(Doctor doctor), insert(Admin)) which put a user with his data in the database and return the result of this action, 3 validate(Class obj) functions (validate(Patient patient), validate(Doctor doctor), validate(Admin admin)) which are used to validate the data given at the login form for each type of user. There is also an availabilityEntry(Appointment app) function in the Dao.class file which's functionality is for each doctor to set the specific dates and hours that he can examine a patient, so a patient can later book an appointment with the doctor

Servlets: This package contains the servlet classes used for the dynamic web app. Servlets are Java classes that are deployed in the sever to handle web request and responses. This is done with the use of doGet and doPost functions (depending on the request type). This package contains 6 Servlet classes: DoctorRegister.java, PatientRegister.java, AdminRegister.java, LoginServlet.java, AvailabilityEntry.java

LoginServlet.java first instantiates an object of the Dao class for the connection to the database to happen, then takes the parameters of the request ("username", "password") and tries to create a response for the user by checking if the credentials are stored in the database. If the validation of the user's credentials is successful, the user is redirected to one of these pages according to the user's role in the platform: patientUtils.jsp, doctorUtils.jsp, adminUtils.jsp , else the user is redirected to the loginFailed.html page.

AvailabilityEntry.java first instantiates an object of the Dao class for the connection to the database to happen, then the data of the available appointment are added in the database in the available_appointments table.

DoctorRegister.java, PatientRegister.java, AdminRegister.java classes all instantiate an object of the Dao class for the connection to the database to happen, then take the parameters of the request, which are the new user's information and with the help of the Dao object the insert method is called. With this method the user is registered successfully in the database. If no errors occur, the user is redirected to the login.jsp page after a short amount of time.

- **Web Content:**

  META-INF: The META-INF folder is an internal Java meta directory, the files/directories contained inside this directory are recognized and interpreted by the Java platform to configure applications, extensions, class loaders and services. it contains the MANIFEST.MF file that is used to define extension and package related data.

  WEB-INF: It contains all resources needed to run the application, from the web deployment descriptor (the web.xml file), to Java classes, JAR files and libraries, to other supporting files that the developer does not want a web user to access. The most used file in that directory is the web.xml file which can be used to configure the whole web application.

  JSP – CSS FILES: JSP files are server-generated web pages. They are similar to .asp or .php files, but contain Java code. The code is parsed by the web server which generates HTML that is sent to the user's computer. CSS files (cascading style sheets) are used to format the contents of a webpage. In other words they are used to modify HTML elements such as size, color, font etc.

  index.jsp: This is the default page. It redirects the user at login.jsp

  login.jsp: This page takes 2 inputs, a username and a password. When the login button is pressed the value of the "action" variable "request.getContextPath()/LoginServlet" is executed with the "method" variable having the "post" value.

  logout.jsp: This page makes the current session invalid and redirects the user to the login.jsp page

  adminRegistration.jsp, doctorRegistration.jsp, patientRegistration.jsp: These pages are used when a new user wants to create a new entry about his profile at the database. This is done after the user completes and submits the specific form. The value of the "action" variable "request.getContextPath()/(AdminRegister or DoctorRegister or PatientRegister)" is executed with the "method" variable having the "post" value.

  registrationOptions.html: This page is used to redirect the user to the competent registration page according to the role that he wants to have in the platform.

patientData.jsp: This page is used to dispay the user’s data once he is logged in. This is done after a connection with the database is established and an sql query is performed. The same process is occurring when the page is also used to display the history of past appointments of each logged in user.

patientUtils.jsp: This page offers a patient the option to book an appointment or manage it.

patientBook.jsp: Sends the data to patientBookII.jsp (Doctor Name, Doctor Specialty etc…)

patientBookII.jsp: Receives the data from patientBook.jsp and makes the mandatory changes in the database. In other words, it deletes the entry from the available_appointments and creates a new entry at the arranged_appointments table .

doctorUtils.jsp: This page offers a doctor the option to enter his availability or manage his appointments.

availabilityEntry.jsp: In this page each Doctor can enter his availability through a form so the data gets parsed by the AvailabilityEntry.java servlet which then, via the Dao class the available appointment is stored in the database.

adminUtils.jsp: This page offers an admin the option to add a patient, to add a doctor or delete a doctor.

deleteDoctor.jsp: This page is used by administrator users in order to delete Doctors from the platform with the help of deleteDoctorII.jsp

deleteDoctorII.jsp: This page is used to delete the selected Doctor from the javabasics.doctor and javabasics.users database tables.

doctorManage.jsp: This page is used for each doctor to see his arranged appointments and manage them.

doctorManageII.jsp: This page gives the functionality of an appointment to be canceled by a doctor and therefore delete it from the javabasics.arranged_appointments database table.

patientManage.jsp: .jsp page which shows the user the future and past appointments he has arranged.

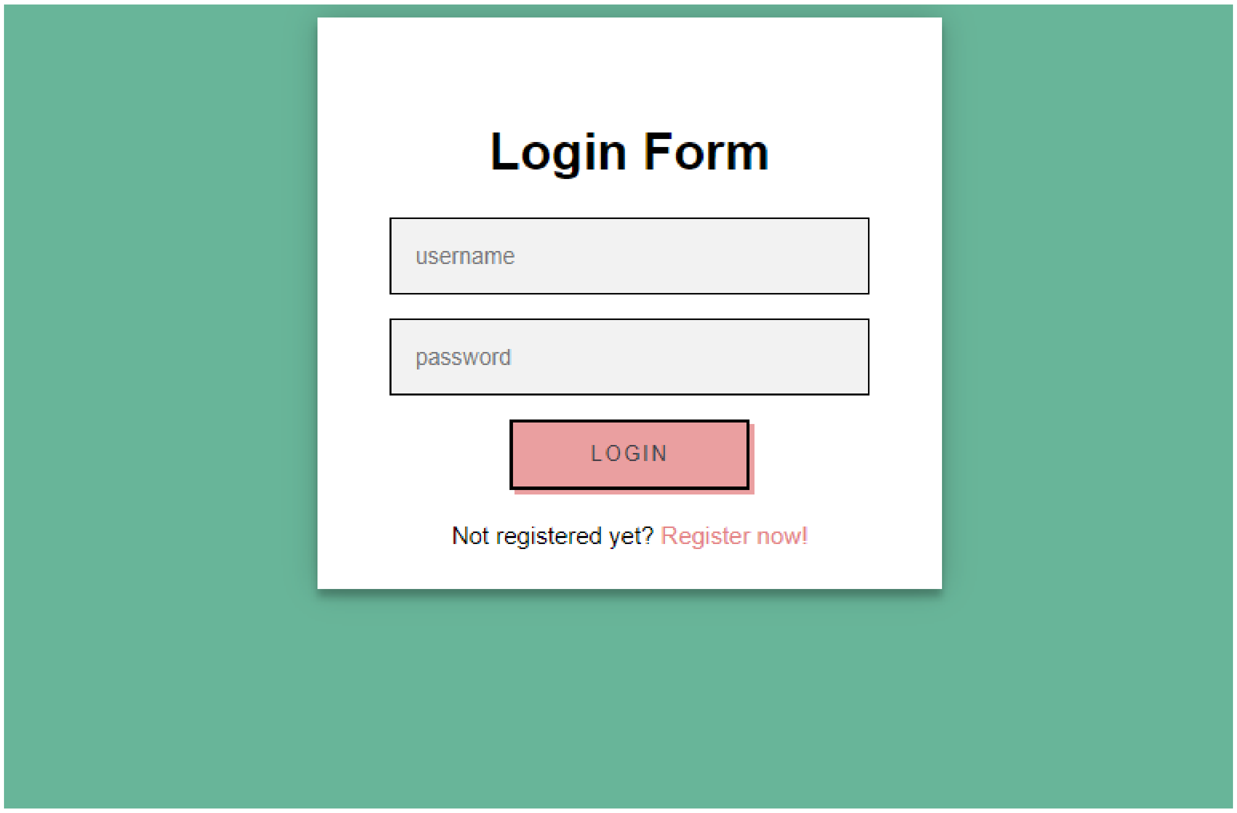
patientManageII.jsp: .jsp page which works with patientManage.jsp and gives the user the ability to cancel one of his future appointments if it’s arranged for later than 3 days.

- **Server:**

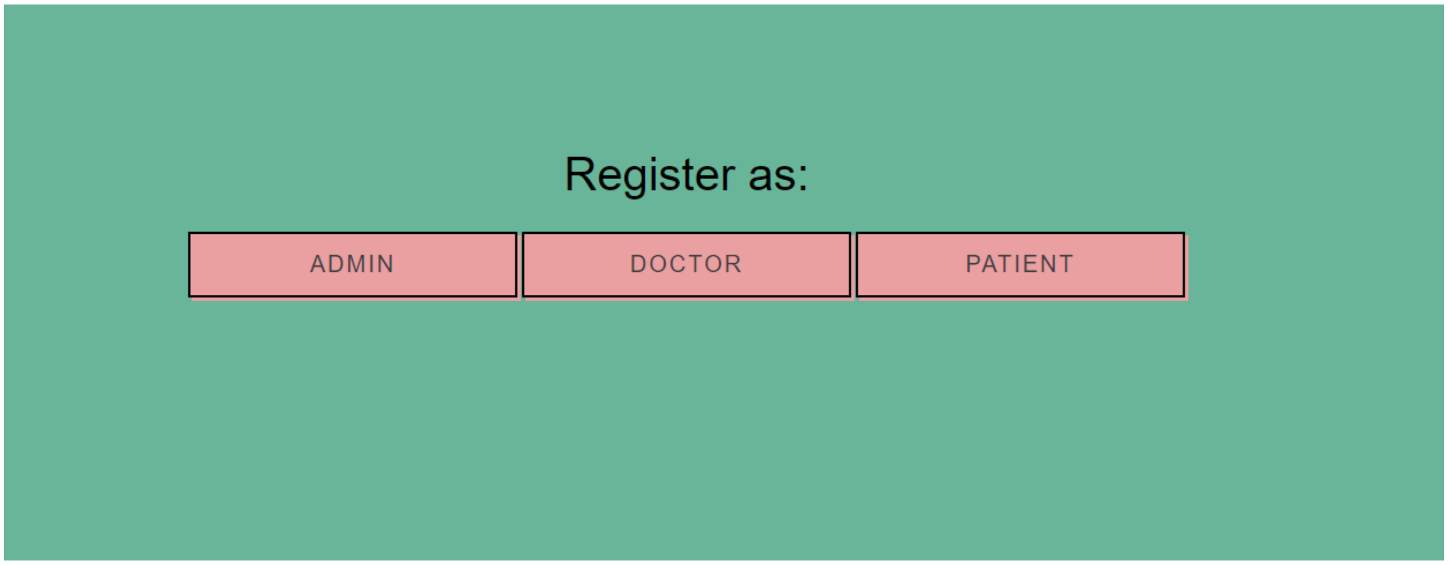
  Tomcat Server at localhost with the HTTP/1.1 port set at port 8080

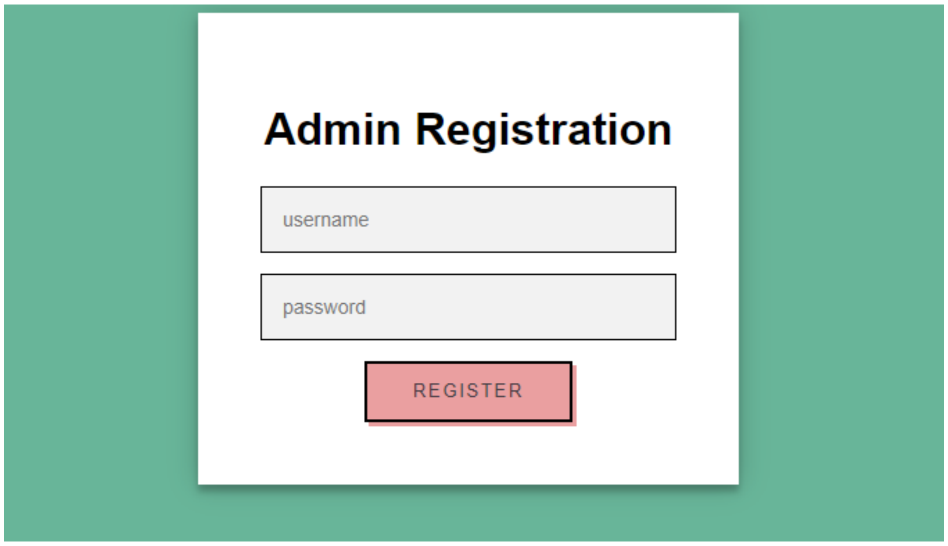
Execution Examples:

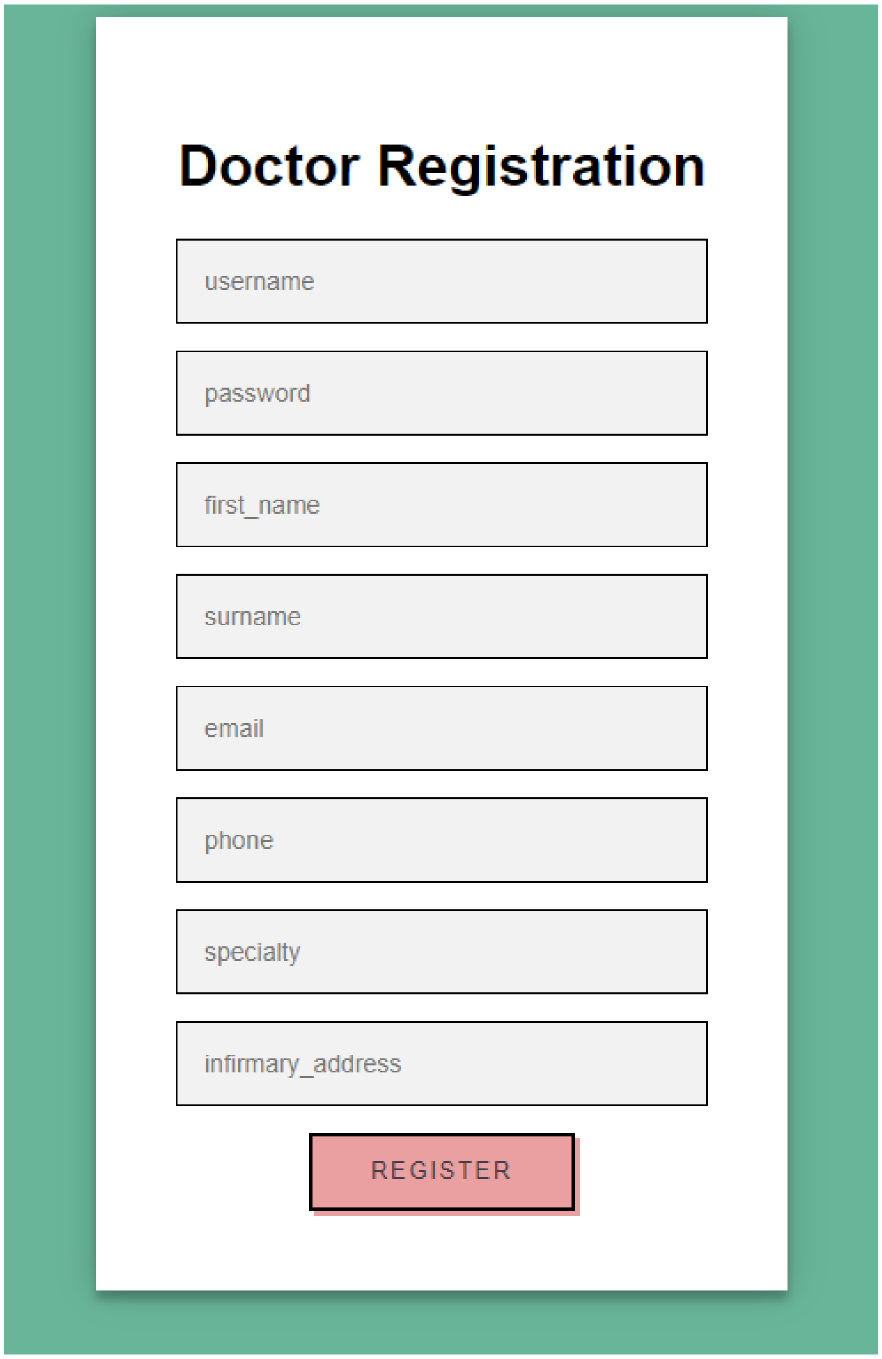
Once the user runs the application the below login form appears. The user can either login entering his username and password or if he is not registered yet, pressing the “Register now!” label he can register in the system.

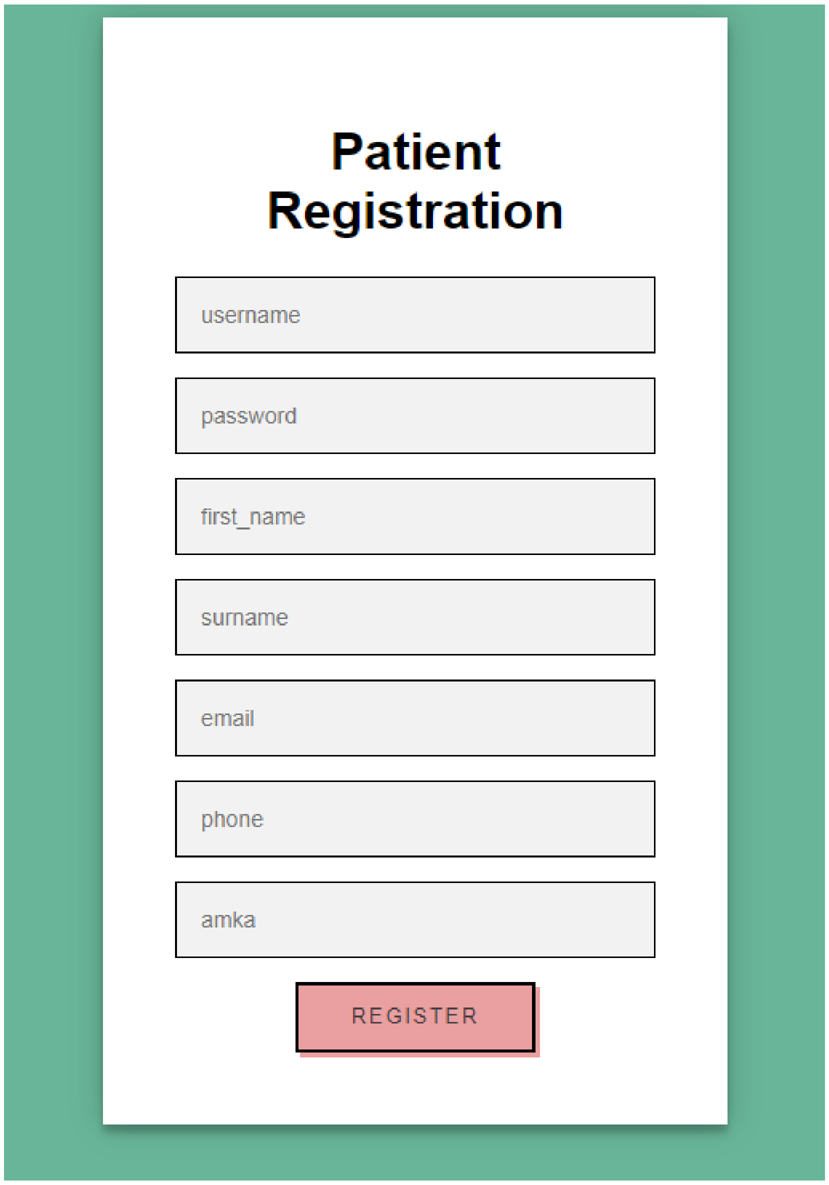


If the user chooses to register, the below form appears. The user has the option to register as an admin, doctor or patient. Pressing any of the buttons a different registration form appears asking for different user input based on the role chosen.



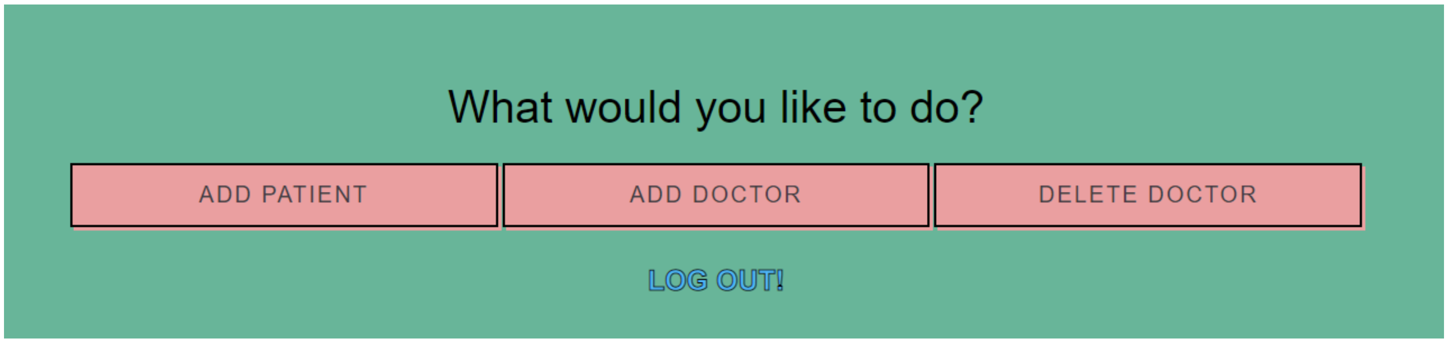

Admin Registration


Doctor Registration

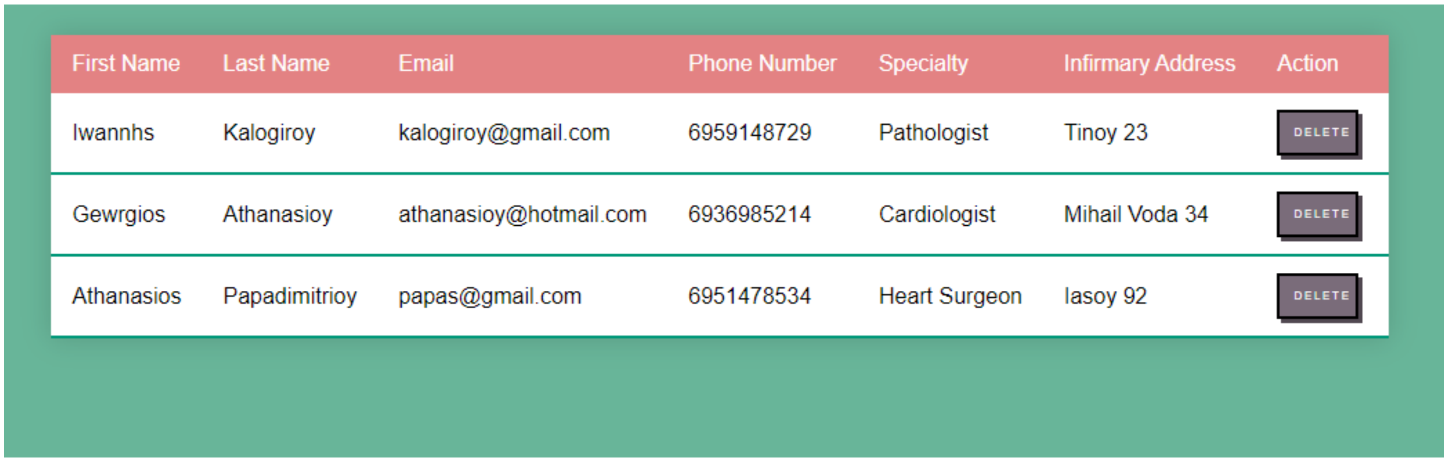


During the login, the system checks what is the role of the user who logged in. Connecting to the database and validating the user input finds if the user who logged in is a doctor, a patient or an admin and then redirects him to the corresponding form.

Connecting as an admin (username: admin, password: 123456):

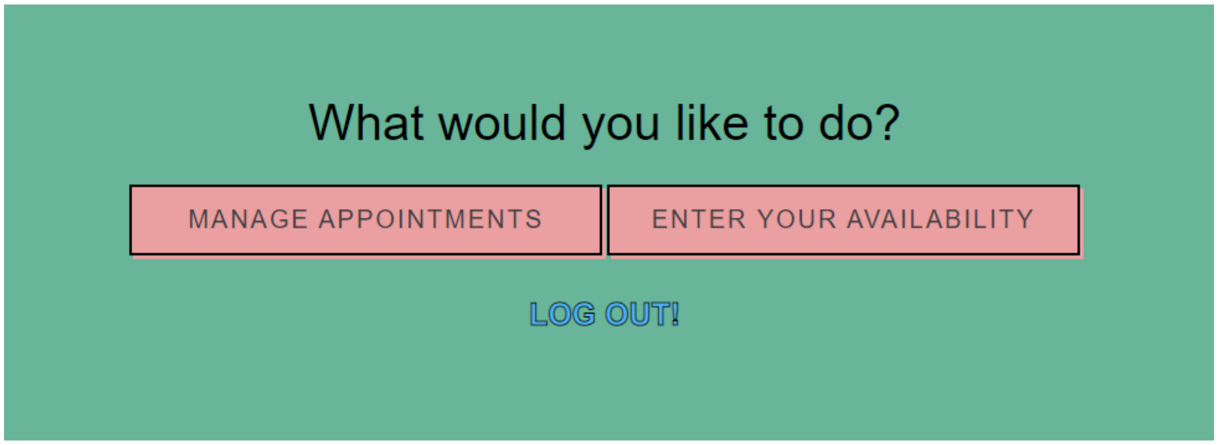


The administrator has the option to add a patient or a doctor and delete a doctor. Pressing the delete doctor button, the registered doctors appear and the admin can delete any of them pressing the delete button.



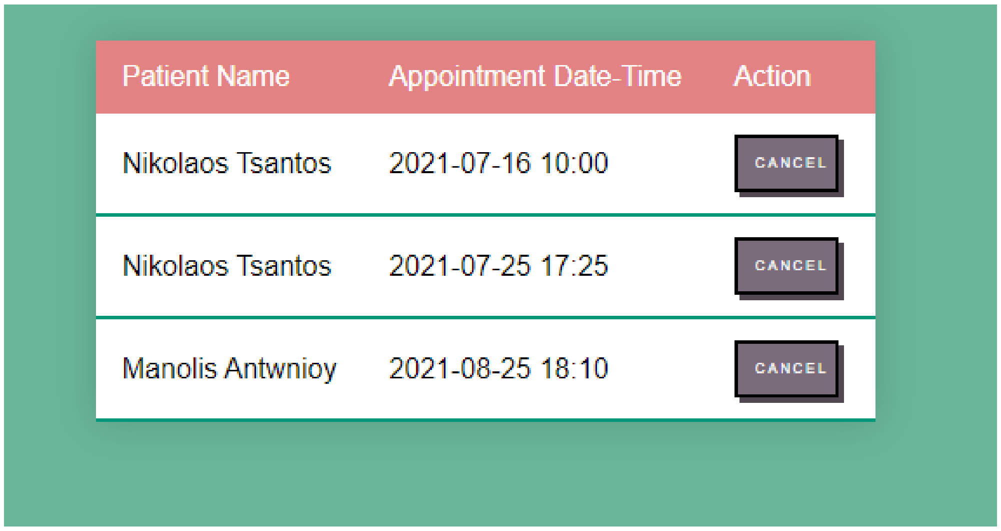
First Name	Last Name	Email	Phone Number	Specialty	Infirmary Address	Action
Iwannhs	Kalogiroy	kalogiroy@gmail.com	6959148729	Pathologist	Tinoy 23	DELETE
Gewrgios	Athanasioy	athanasioy@hotmail.com	6936985214	Cardiologist	Mihail Voda 34	DELETE
Athanasios	Papadimitrioy	papas@gmail.com	6951478534	Heart Surgeon	Iasoy 92	DELETE

Connecting as a doctor (username: doctor/ doctor2/ doctor3, password: 123456): The doctor can either manage his appointments or enter his availability.



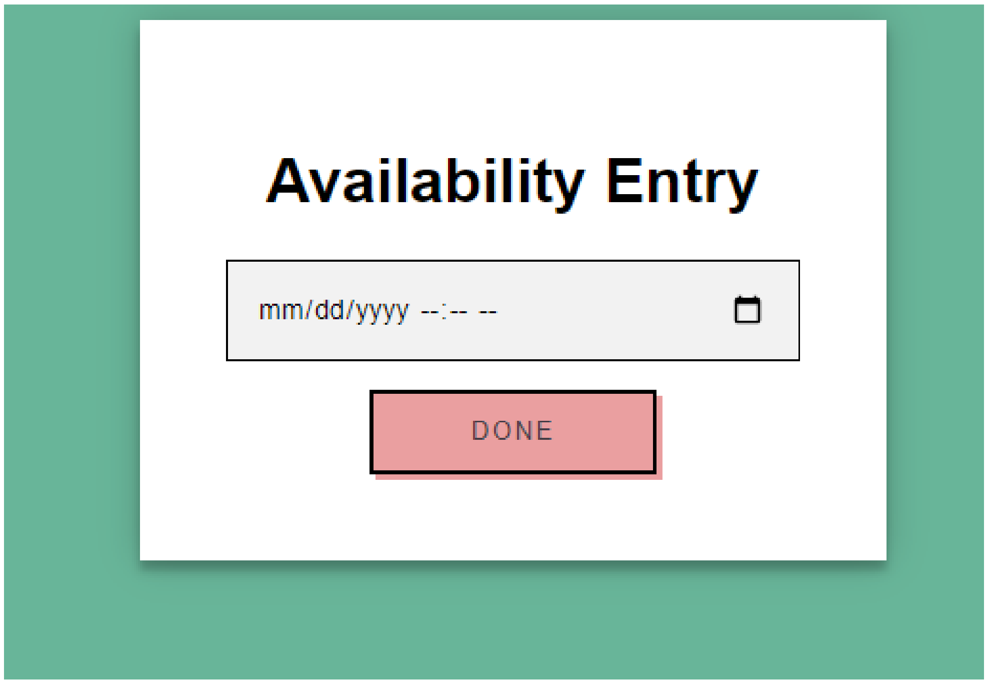
What would you like to do?

Manage appointments: the arranged appointments for that doctor appear and he has the option to cancel them if they are later than three days.

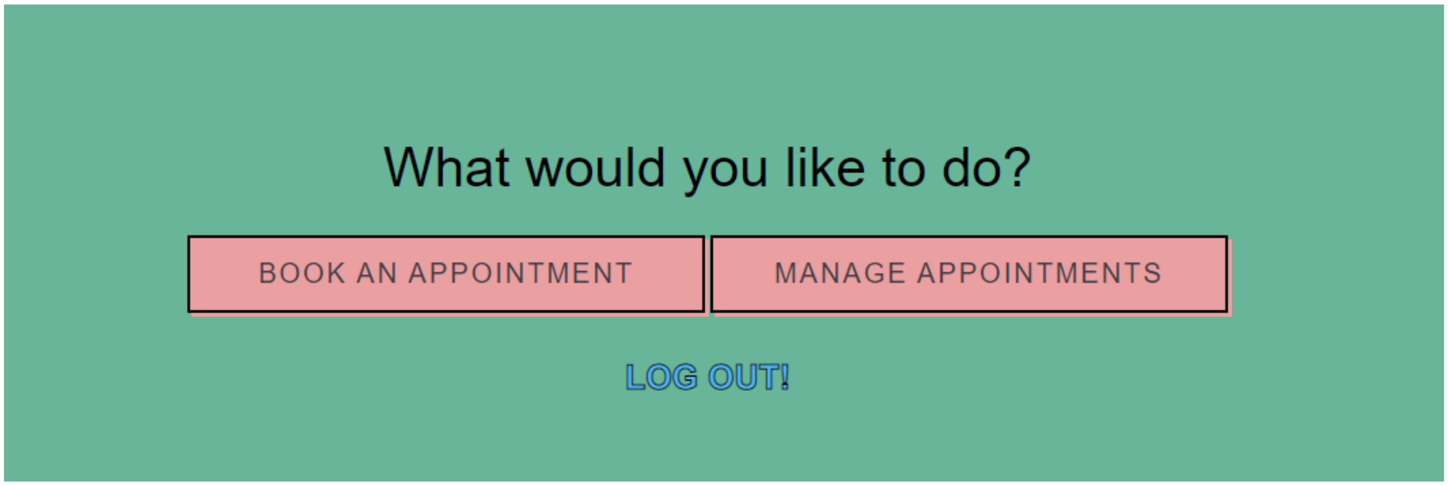


Patient Name	Appointment Date-Time	Action
Nikolaos Tsantos	2021-07-16 10:00	CANCEL
Nikolaos Tsantos	2021-07-25 17:25	CANCEL
Manolis Antwnioy	2021-08-25 18:10	CANCEL

Enter your availability: a calendar appears with which the doctor can choose his availability hours and dates.

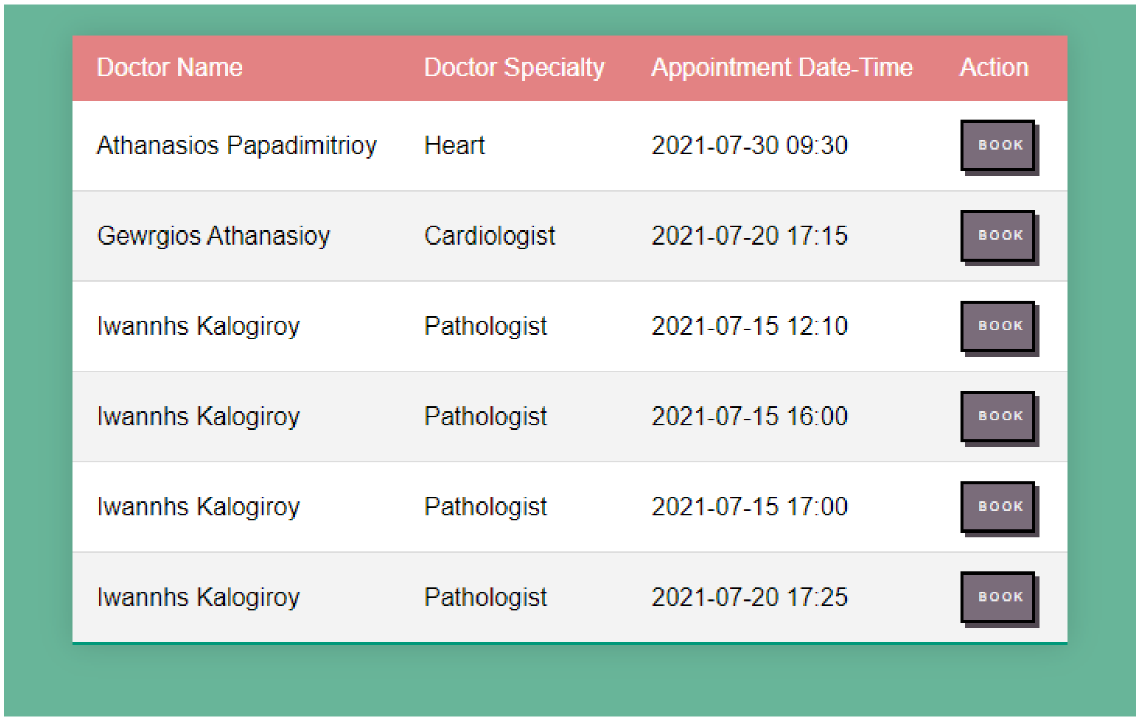


Connecting as a patient (username: patient/ patient2, password: 123456):



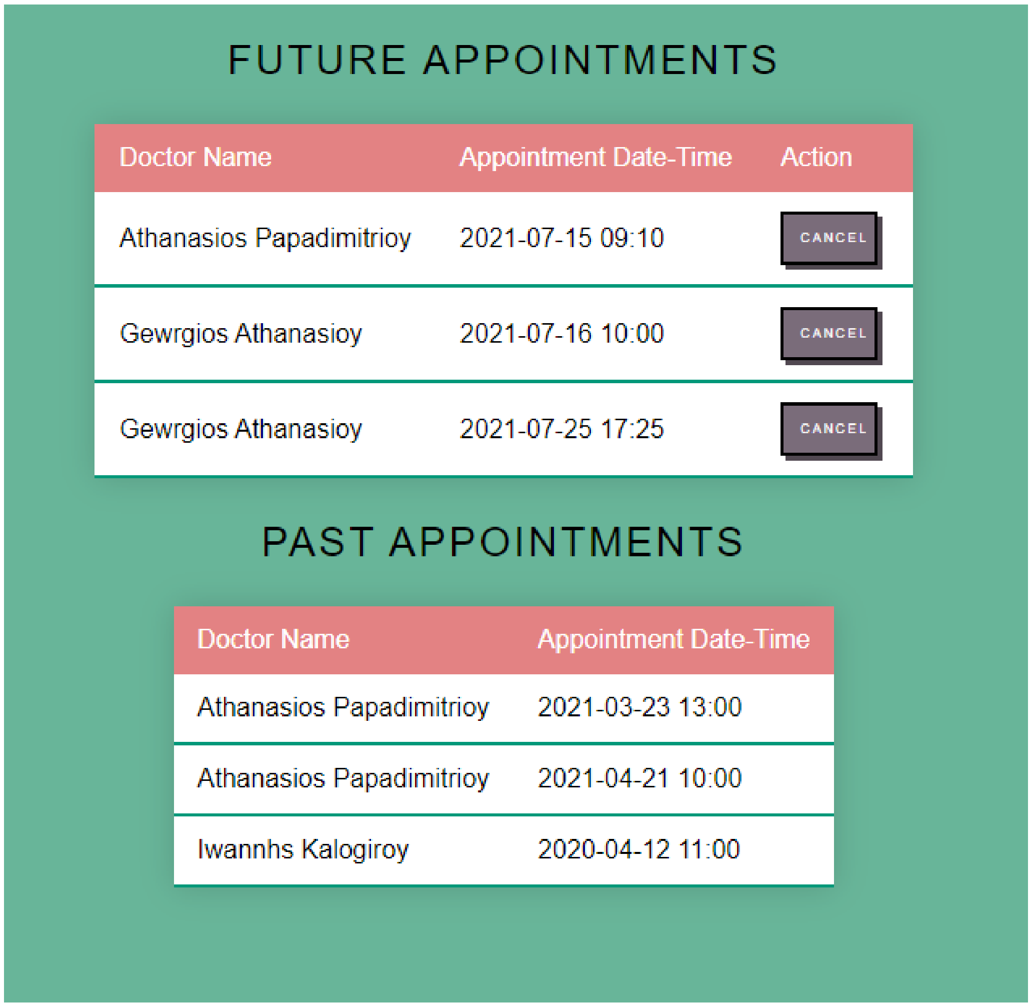
The patient can book an appointment or manage his arranged appointments pressing the corresponding button.

Book an appointment: the available future appointments appear and the user can choose to book one of them pressing the book button.



Doctor Name	Doctor Specialty	Appointment Date-Time	Action
Athanasios Papadimitrioy	Heart	2021-07-30 09:30	BOOK
Gewrgios Athanasioy	Cardiologist	2021-07-20 17:15	BOOK
Iwannhs Kalogiroy	Pathologist	2021-07-15 12:10	BOOK
Iwannhs Kalogiroy	Pathologist	2021-07-15 16:00	BOOK
Iwannhs Kalogiroy	Pathologist	2021-07-15 17:00	BOOK
Iwannhs Kalogiroy	Pathologist	2021-07-20 17:25	BOOK

Manage appointments: two tables appear, one that contains the future and one that contains the past appointments for the user based on the date he has connected. The user has the option to cancel any of his future appointments if it's later than three days.



FUTURE APPOINTMENTS

Doctor Name	Appointment Date-Time	Action
Athanasios Papadimitrioy	2021-07-15 09:10	CANCEL
Gewrgios Athanasioy	2021-07-16 10:00	CANCEL
Gewrgios Athanasioy	2021-07-25 17:25	CANCEL

PAST APPOINTMENTS

Doctor Name	Appointment Date-Time
Athanasios Papadimitrioy	2021-03-23 13:00
Athanasios Papadimitrioy	2021-04-21 10:00
Iwannhs Kalogiroy	2020-04-12 11:00