FindMyPatient V2

(Online extension)

Gerard Klomphaar

Table of Contents

Subject description

Goal

This project is created as a mean to show sufficient knowledge of java which is acquired during the Advanced Java classes at EPITA.

During these java classes a range of technologies and best practices related to java is shown, this knowledge is implemented in this project to show sufficient knowledge about the technology and how to apply them in practice.

Use case

As a use case a hospital patient management system is created which includes all the technologies and best practices acquired during java class. This system is an extension of the developed application during the previous java class.

The hospital management system which is created should be a web application. The main features are analyzed in the next chapter.

Subject analysis

The minimal requirements for the system are extracted from the specification provided on the Advanced Java webpage. The requirements are applied to the use case.

Major features

The following 4 categories of features and associated requirements are extracted:

Main application features

REQ 1	Login page for authentication
REQ 2	Welcome page, as a sort of dashboard
REQ 3	Creation page
REQ 4	Search page • Delete option • Modify option
REQ 5	Use Hibernate or JDBC to store data models
REQ 6	Use Junit to test
REQ 7	Scalability
REQ 8	Make it possible to dynamically add new fields to the data model

Application Feasibility

To determine the feasibility we will analyze the requirements and determine the risk for the realization of this requirement

Table 1. Requirement analysis

Requirement	**Analysis**	**Risk**
REQ1 [3], *REQ2 [3]*, *REQ3 [3]*, *REQ4 [3]*	The creation of webpages including the communication with servlets and backend processing is covered during class and can used in this project.	Low risk
REQ5 [3]	The data models are simple and Hibernate and JDBC are known technologies.	Low risk
REQ6 [3]	The context in this project is different from the learned content during class. Since we are working with Spring and JUnit on servlets this could provide issues while testing and can result in addition time spend.	Medium risk
REQ7 [3]	Scalability in the context of web development is a new area which knowledge needs to be accumulated, this can result in potential extra work	Medium risk
REQ8 [3]	The addition of new fields in a web application is an	High risk

Requirement	Analysis	Risk
	unknown area, since the data models are static, additional technologies or methods need to be researched. This will probably result in lots of effort.	

Data description

The project mainly manages patient data, additional there is User data which will be used for authentication. Below the two data types are displayed.

Table 2. Patient data structure

Patient	
Field	**Type**
id	int
First name	String
Last name	String
Social Security number	String
Telephone number	String
Email	String
Room	String

Table 3. User data structure

User	
Field	Type
Id	int
Name	String
Password	String
Rights	int

Expected results

The result of this project should contain the features and extracted requirements. There should be documentation which is clear enough to get the main idea of the application and its structure.

Algorithms study

The different possibilities to realize certain functions are analyzed to result in the best applicable solution.

[A1] Saving data

To the two data models User and Patient previously a specific DAO implementation is used. This implementation could handle different data models, but cannot handle dynamic data. Since the application should be scalable *REQ7 [3]*, it is preferred to have a more dynamic solution. Hibernate can be a better solution to provide this *REQ5 [3]*.

[A2] Searching data

The application should feature a search method. Since the data should be scalable *REQ7 [3]* and *REQ8 [3]*, the search should be able to handle dynamic fields.

[A3] Website Responsiveness

It should be able to do operations on Patients like create, modify and delete without unnecessary page reloads.

[A4] Packaging & Architecture

The different features required should be organized in a way that is maintainable. The basic functionality from the previous java semester can be used where necessary.

[A5] Authorization & Security

To provide security an authorization mechanism should be provided. The mechanism of the previous project will be used. A method should be created to save the User credentials and status for each connected in parallel.

[A6] Software configuration management

To control the different version of the used packages a system needs to be used.

Scope of the application

The application is restricted to the features specified in chapter Major features. Additional features which are out of scope:

- The graphical design of the website (basics representation be implemented).
- Additional security: e.g. encryption of the http session, encryption of saved User data.

Conception

This chapter provides the chosen design for the application.

Chosen algorithm

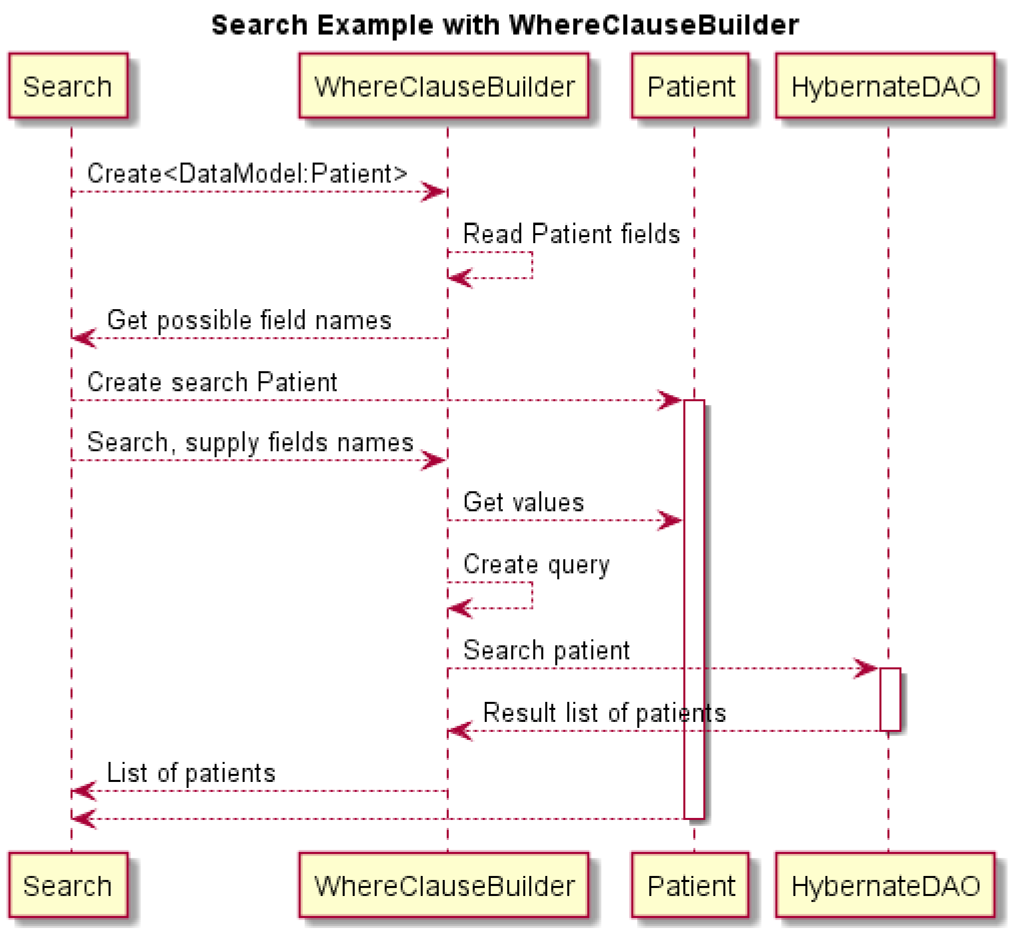
[A1] Saving data

Saving data is done using the hybernate technology, this ensures the flexibility required.

[A2] Searching data

To search data based on the potential dynamic fields of the data models, a custom where clause generation feature will be created. This clause can then be used with the chosen hybernate technology to execute the search. In *Figure 1, "Search example using WhereClauseBuilder"* below the sequence of searching using the "WhereClauseBuilder" is depicted.

Figure 1. Search example using WhereClauseBuilder



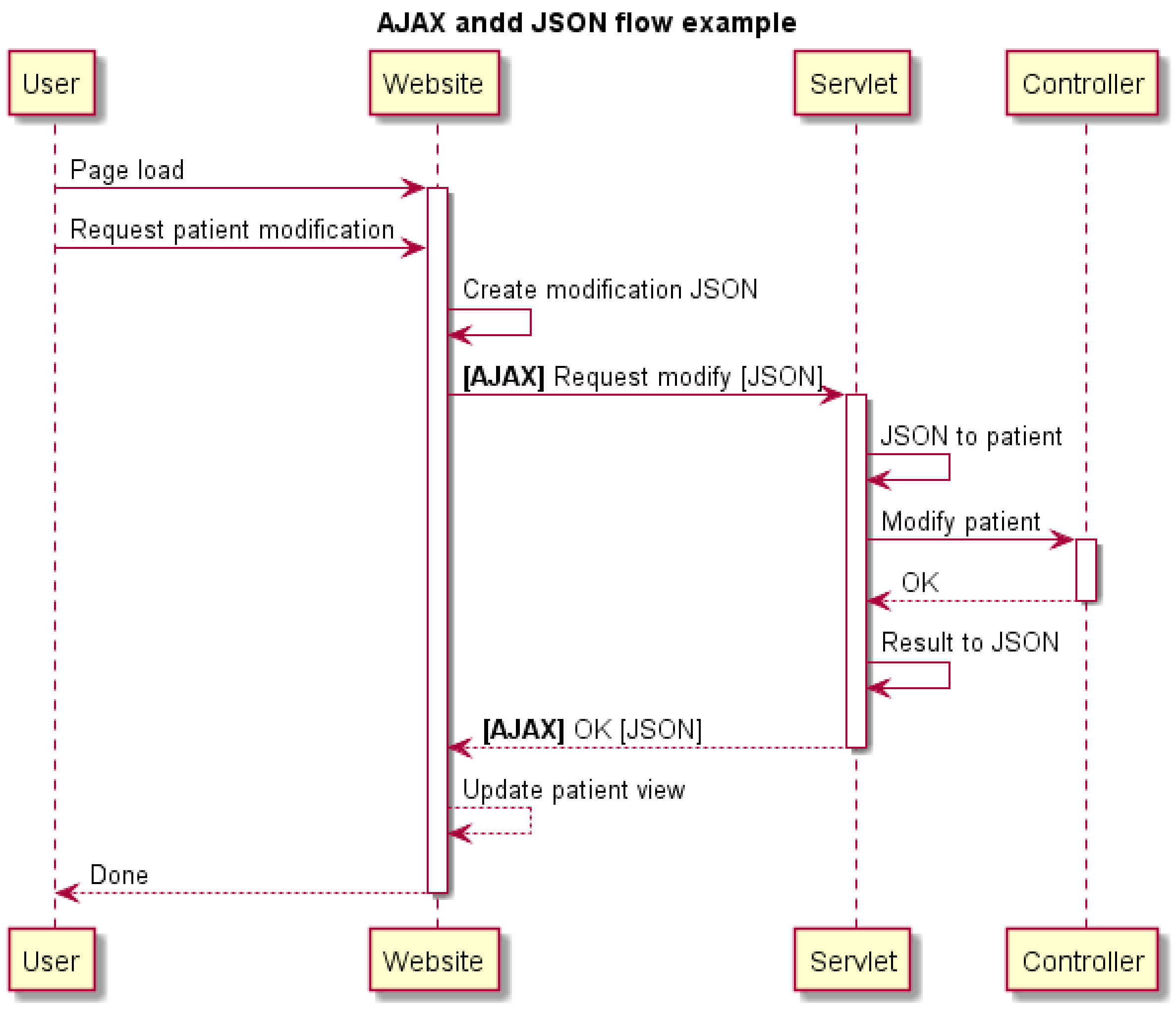
[A3] Website Responsiveness

To provide a responsive flow to the user without unnecessary refreshes of pages the AJAX technology will be used.

Sending data between Pages and servlets will be done using JSON objects. This provides an efficient and scalable solution for data transfer.

Figure 2, “AJAX and JSON flow example” depicts the sequence which is used for modifying a Patient.

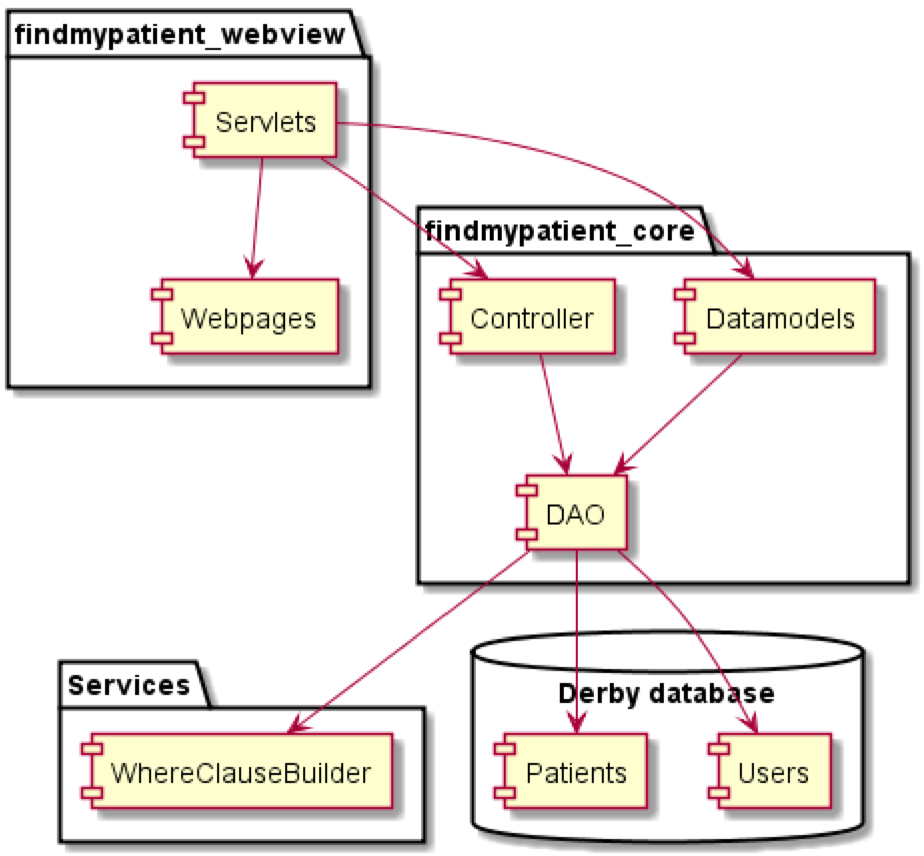
Figure 2. AJAX and JSON flow example



[A4] Packaging & Architecture

To organize the different features the project is subdivided in the packages as shown in *Figure 3, “Packaging overview”*, each package is a project.

Figure 3. Packaging overview



[A5] Authorization & Security

To provide authorization for each individual connected user each user will have an individual Controller assigned. The controller which also has the authentication functionality is saved in the http session with the user.

To provide additional security, no additional content or pages can be accessed without logging in. Any post or get functionality will not work without a login.

A future improvement would be to encrypt the HTTP session and to encrypt the password which is currently saved as a raw string in the database.

[A6] Software configuration management

Maven

To provide a configuration environment Maven is used. The project will comply with the company versions specified in the company Maven “super POM”.

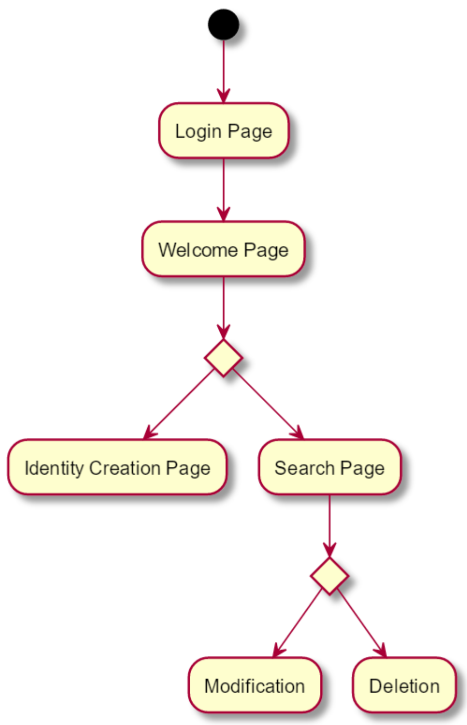
Spring

To provide a flexible software application the dependency injection framework Spring is used. Interfaces will be created to be able to inject different implementations.

Global application flow

The global application flow is based on the specified requirements (*REQ1 [3]REQ2 [3]REQ3 [3]REQ4 [3]*). The flow is depicted in *Figure 4, “Application flow”*

Figure 4. Application flow

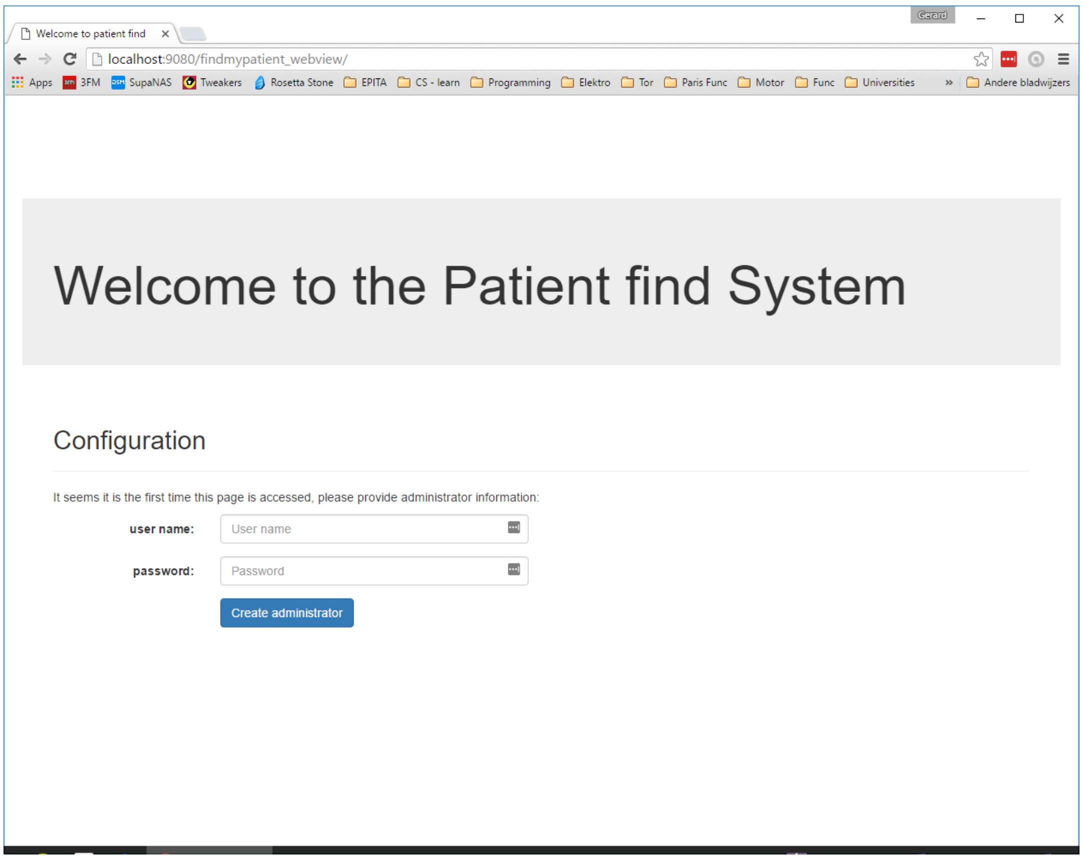


GUI description

LOGIN

At first access to the webpage the backend database will still be empty. This will result in the display of a configuration page as shown in *Figure 5*, *“Configuration page”*. Through this page the initial administrator can be created. After creation any following access will display the login page Figure 2.

Figure 5. Configuration page



Welcome to the Patient find System

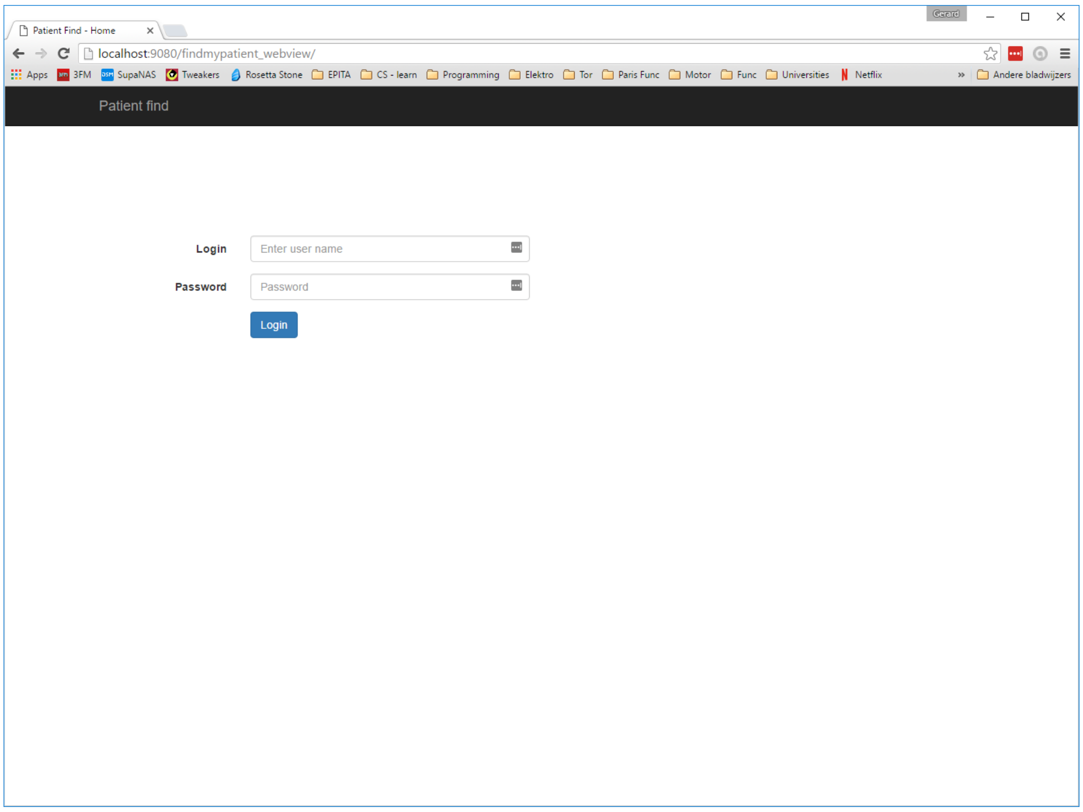
Configuration

It seems it is the first time this page is accessed, please provide administrator information:

user name:	User name
password:	Password

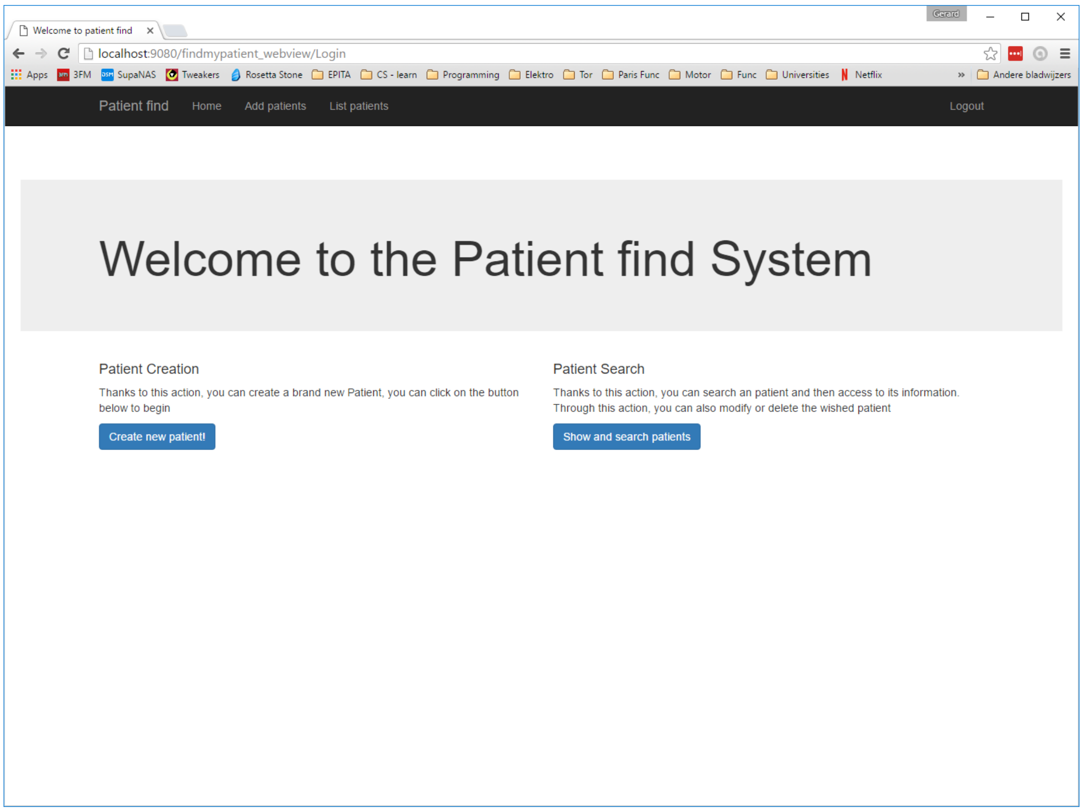
Create administrator

Figure 6. Login page



After login a welcome page will be displayed as shown in *Figure 7, “Welcome page”*, which gives access to creating new patients or show the current patients. At the top of the screen the navigation bar can also be used. At the top right the logout option is can be found.

Figure 7. Welcome page



Welcome to the Patient find System

Patient Creation

Thanks to this action, you can create a brand new Patient, you can click on the button below to begin

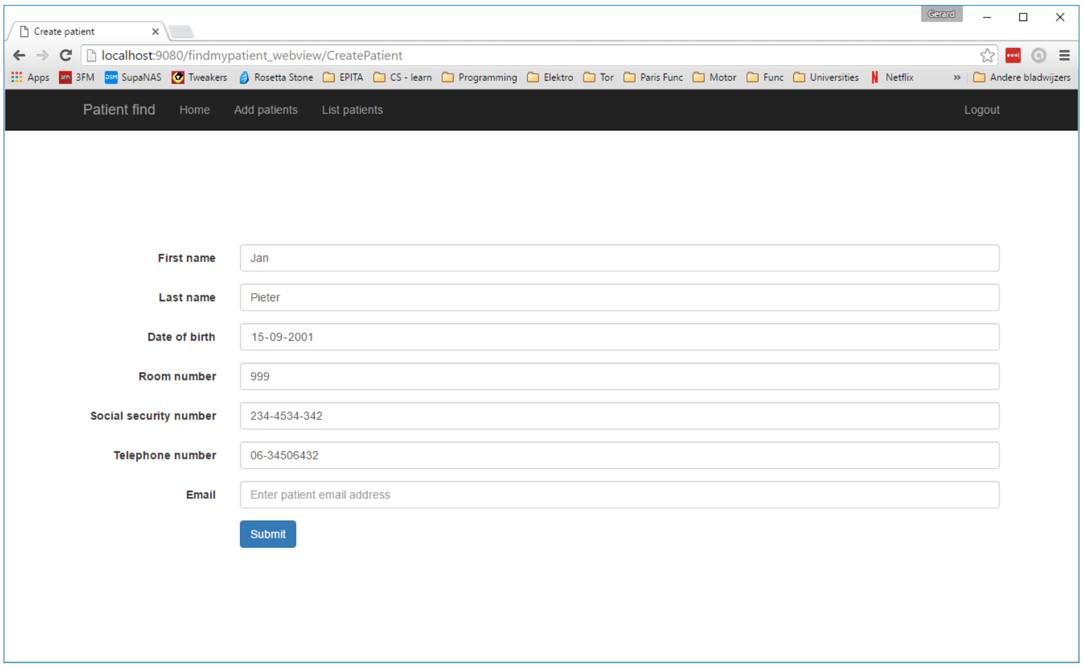
Create new patient!

Patient Search

Thanks to this action, you can search an patient and then access to its information. Through this action, you can also modify or delete the wished patient

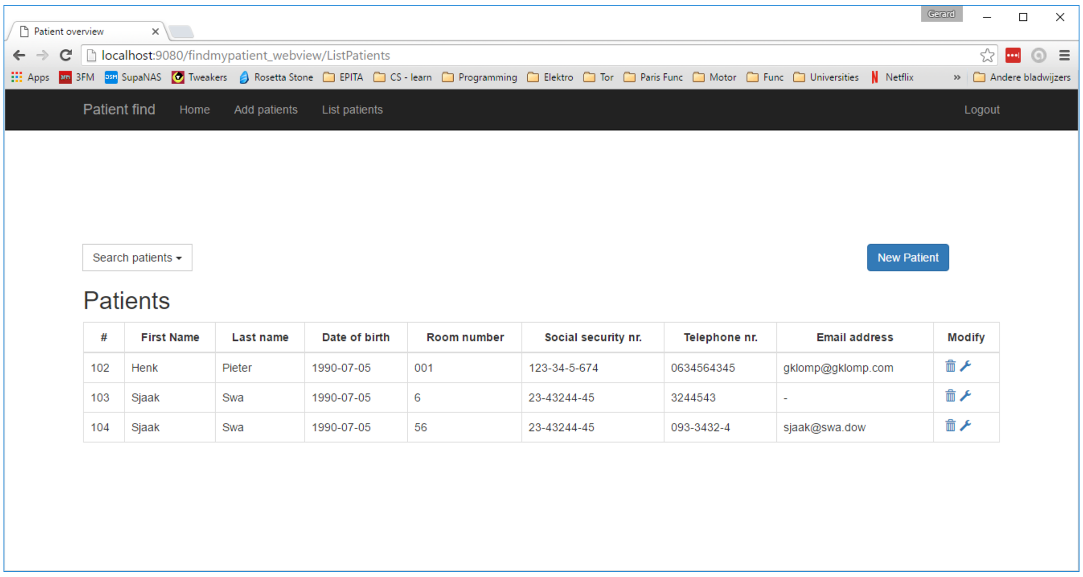
Show and search patients

Figure 8. Create patient page



First name	Jan
Last name	Pieter
Date of birth	15-09-2001
Room number	999
Social security number	234-4534-342
Telephone number	06-34506432
Email	Enter patient email address

Figure 9. Patient overview page

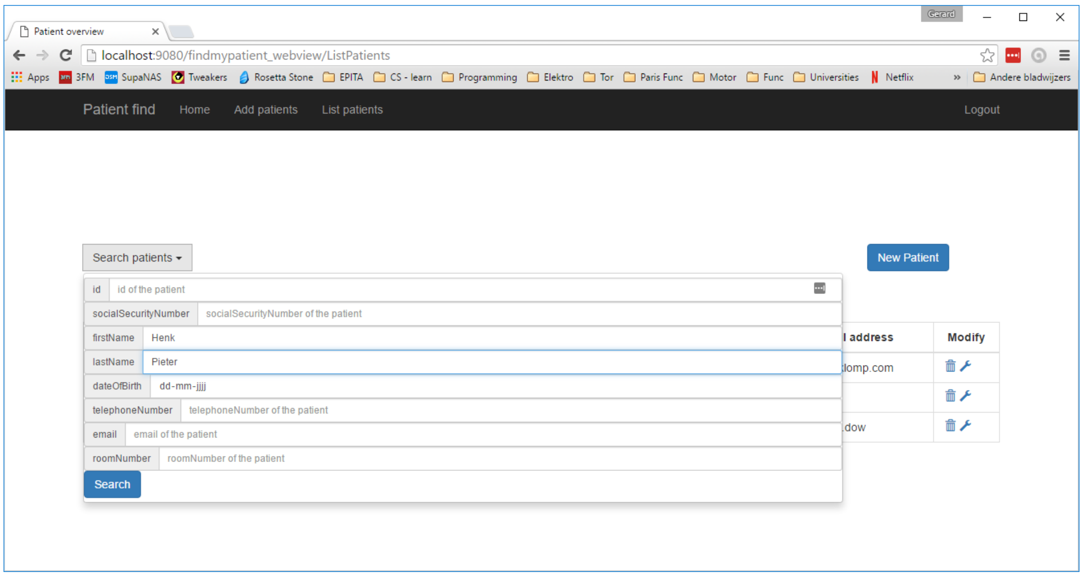


Patients

#	First Name	Last name	Date of birth	Room number	Social security nr.	Telephone nr.	Email address	Modify
102	Henk	Pieter	1990-07-05	001	123-34-5-674	0634564345	gklomp@gklomp.com	
103	Sjaak	Swa	1990-07-05	6	23-43244-45	3244543	-	
104	Sjaak	Swa	1990-07-05	56	23-43244-45	093-3432-4	sjaak@swa.dow	

In the “list patients” page search actions can be executes. Multiple parameters can be combined in a single search.

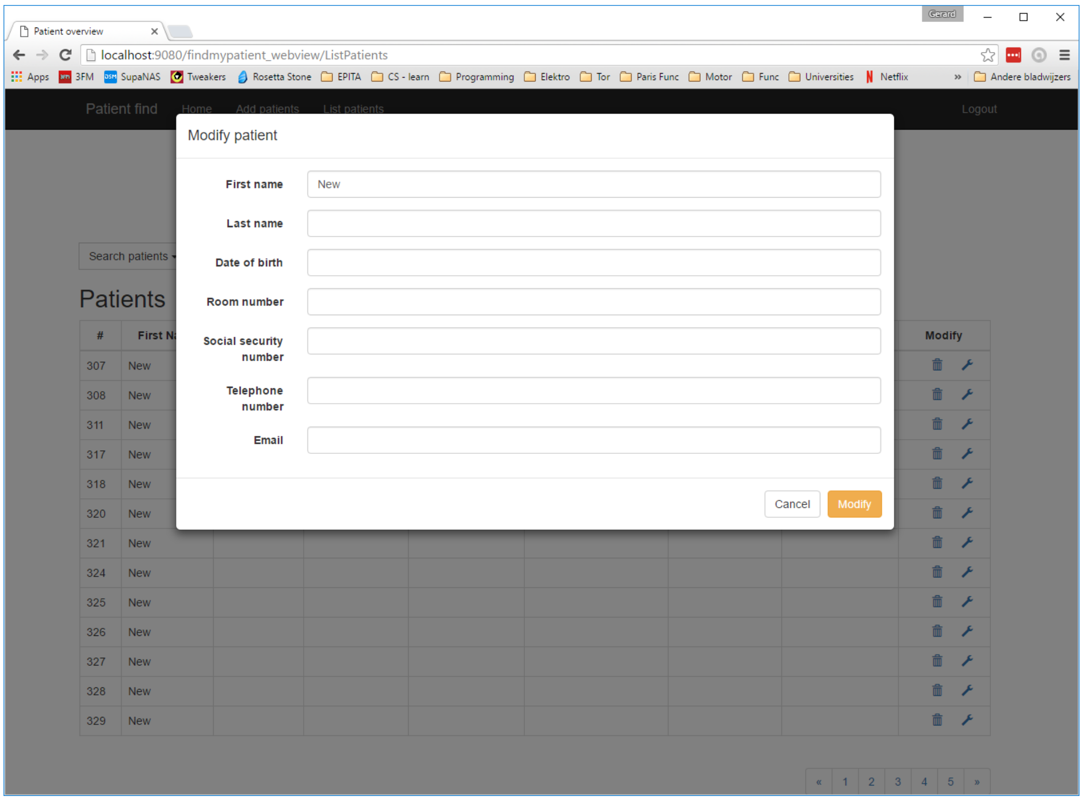
Figure 10. Search patient page



id	id of the patient
socialSecurityNumber	socialSecurityNumber of the patient
firstName	Henk
lastName	Pieter
dateOfBirth	dd-mm-jjjj
telephoneNumber	telephoneNumber of the patient
email	email of the patient
roomNumber	roomNumber of the patient

At the right of each patient in the patient overview list 2 actions are displayed, delete and modify. Both will pop up a new window in which the actions can be confirmed. see *Figure 11, “Modify patient action”* and *Figure 12, “Delete patient action”*

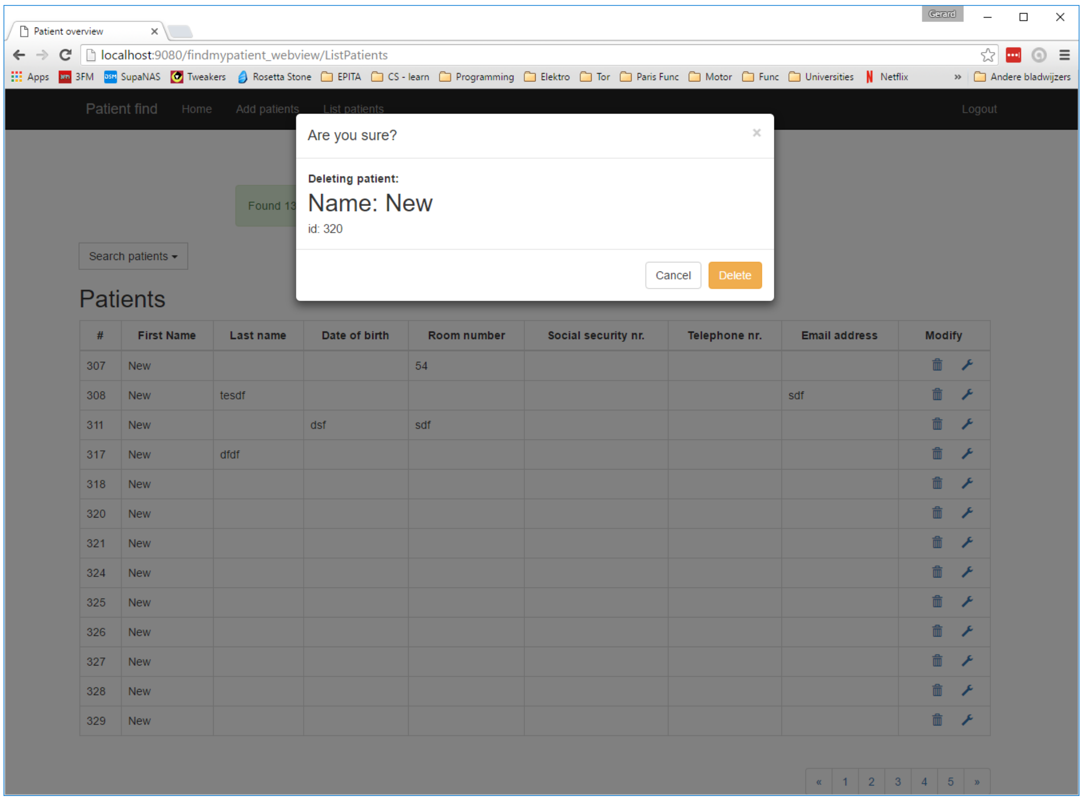
Figure 11. Modify patient action


Patient overview
localhost:9080/findmypatient_webview/ListPatients
Apps 3FM SupaNAS Tweakers Rosetta Stone EPITA CS - learn Programming Elektro Tor Paris Func Motor Func Universities Netflix Andere bladwijzers
Patient find Home Add patients List patients Logout
Search patients
Cancel
Modify
« 1 2 3 4 5 »

Modify patient

First name	New
Last name	
Date of birth	
Room number	
Social security number	
Telephone number	
Email	

Figure 12. Delete patient action


Patient overview
localhost:9080/findmypatient_webview/ListPatients
Apps 3FM SupaNAS Tweakers Rosetta Stone EPITA CS - learn Programming Elektro Tor Paris Func Motor Func Universities Netflix Andere bladwijzers
Patient find Home Add patients List patients Logout
Found 13
Search patients
Cancel
Delete
« 1 2 3 4 5 »

Are you sure?

Deleting patient:

Name: New

id: 320

Patients

#	First Name	Last name	Date of birth	Room number	Social security nr.	Telephone nr.	Email address	Modify
307	New			54				
308	New	tesdf					sdf	
311	New		dsf	sdf				
317	New	dfdf						
318	New							
320	New							
321	New							
324	New							
325	New							
326	New							
327	New							
328	New							
329	New							

Configuration instructions

Database configuration

The database is run as a derby in memory database. It is a persistence version and the location is: ‘PatientsDB’ (`url:"jdbc:derby:PatientsDB;create=true"`). It is tested with a Tomcat v7.0 server. If preferred above url can be changed in ”applicationContext.html”.

The **PatientsDB** part can be replaced with an absolute path, e.g. `"jdbc:derby:C:/PatientsDB;create=true"`).

Open in browser

The website is generally run on: http://localhost:”configuratedPort”/findmypatient_webview