Insert here your thesis' task.

CZECH TECHNICAL UNIVERSITY IN PRAGUE

FACULTY OF INFORMATION TECHNOLOGY

DEPARTMENT OF SOFTWARE ENGINEERING

Bachelor's thesis

Specialized Information System Maintaining Patients Participating in Epileptosurgical Programme – Reporting Module

Martin Dvořáček

Supervisor: Ing. Petr Ježdík Ph.D.

13th May 2014

Acknowledgements

This thesis has benefited greatly from the support of many people, some of whom I would sincerely like to thank here.

To begin with, I am deeply grateful for the support and guidance of Mr. Ježdík with whom I have a possibility to cooperate for past two years. Thanks to him, I could work on such an interesting topic as an implementation of an information system for a the biggest hospital in the Czech Republic.

Also, I owe special thanks to all my classmates that have somehow participated in the process of development of the GENEPI IS.

My sincere gratitude goes to Tim Rogers for the proofreading of my thesis.

Finally, but first in my heart, my parents and my brother are due my deep gratitude for their continued moral and financial support throughout my studies. The broad education that I was able to enjoy while growing up has proven invaluable.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended, in particular that the Czech Technical University in Prague has the right to conclude a license agreement on the utilization of this thesis as a school work under the provisions of Article 60(1) of the Act.

In Prague on 13th May 2014   . . . .

Citation of this thesis

Dvořáček, Martin. *Specialized Information System Maintaining Patients Participating in Epileptosurgical Programme – Reporting Module.* Bachelor's thesis. Czech Technical University in Prague, Faculty of Information Technology, 2014.

Abstract

Epilepsy is a brain disorder that affects approximately 1% of the population. Of these, approximately 10% of the patients suffer from a pharmacoresistant form of epilepsy which can not be treaten by the standard medical procedures. This paper deals with the analysis, design and implementation of a reporting module to the GENEPI Information System, which was developed for the Faculty Hospital Motol by the students of FIT ČVUT, and is a key tool for the management of patients, involved in epileptosurgical programme. Currently in the epileptosurgical programme are involved 150 patients, mostly children, and its beginning dates back to 2000. GENEPI IS, and thus the reporting module, is software that was developed from the beginning according to the specific requirements of doctors who use it. Given that it stores unique information, which also may contain sensitive information, the resultant product was developed under special claims. Among those included robustness, reliability and safety aspects.

Keywords export of data, access acording to the user roles, data anonymization, reporting module

Abstrakt

Epilepsie je onemocnění mozku, které postihuje přibližně 1% populace. Z toho přibližně 10% pacientů trpí farmakorezistentní formou epilepsie, při které se nedají použít standardní léčebné postupy. Tato práce se zabývá analýzou, designem a implementací reportovacího modulu do Informačního Systému GENEPI, který byl vyvíjen na zakázku pro Fakultní Nemocnici Motol studenty FIT ČVUT, a který je klíčovým nástrojem pro správu těchto pacientů zapojených do epilepticko-chirurgického programu. V současné době je do epilepticko-chirurgického programu zapojeno na 150, převážně dětských, pacientů a jeho začátek se datuje do roku 2000. GENEPI IS, a tím pádem i reportovací modul, je software, který byl od začátku vyvíjen na základě specifických požadavků lékařů, kteří jej používají. Vzhledem k tomu, že uchovává unikátní data, která zároveň mohou obsahovat citlivé údaje, tak byly na výsledný produkt při vývoji kladeny zvlášní nároky. Mezi ty patřila robustnost, spolehlivost a především bezpečnost.

Klíčová slova   export dat, přístup na základě uživatelských rolí, anonymizace dat, reportovací modul

x

Contents

List of Figures

CHAPTER **1**

Introduction

The reporting module, as part of the GENEPI - information system, adds an important functionality to this software. Thanks to this module the user will be able to export data saved in the system to sundry formats. This is useful for the doctors that take care of patients with epilepsy, as well as for the researchers that make analysis above the data from this system. The goal was to extend the IS with additional functionality without letting the user notice he's even using further extension. Therefore the reporting module was delicately worked into the existing source code of the GENEPI IS, using the same design patterns and approaches as the rest of the IS.

The GENEPI IS should replace the original information system that was used in the Faculty Hospital Motol in Prague for maintaining patients in its epileptosurgical programme. The main reasons for replacing the original information system was the fact that it didn't comply with the current requirements of the doctors, contained bugs and its design wasn't optimal. Because of the uniqueness of stored data and because of the special requirements of the system itself, there was no free software available that would be suitable and thus it was needed to create custom information system.

I chose to participate in this project, although I knew it would not be easy. The main reasons that led to selection of participation in the development of the GENEPI IS were the facts that it is a software that will be deployed in the production environment to be used by real users and can help to improve the state of health of real people.

In the following chapters the reader will become familiar with the GENEPI IS, the process of analysis, design and realisation of the reporting module, as well as with the process of testing and will see a few examples of exports done by the reporting module.

CHAPTER $\mathbf{2}$

Current status

That the reader can fully understand the functioning of the reporting module, first of all it is needed to introduce the information system that it extends. The GENEPI information system was being created within the subjects BI-SP1 and BI-SP2 at the Czech Technical University, Faculty of Information Technologies in the school year 2012/2013 as a student project. You will find these projects on enclosed CD.

GENEPI IS is being developed in the java programming language, using frameworks[1] Spring 4.0.2 and Hibernate 4.3.4. The front end part is realized via JSP, using HTLM5, JavaScript and CSS. Libraries that were used to ease the programming of the front end were Twitter Bootstrap 3.3.1 and jQuery 2.1.0. MySQL 5.5 was chosen as a database and Apache Tomcat 7 as an application server. All of the used software and libraries are distributed under some kind of free license. GENEPI IS is a bilingual software due to the expected deployment in the Miami Children's hospital in Florida, the United States of America. Languages that are currently supported, are Czech and English, nevertheless thanks to the suitably implemented localization it is easy to extend the application to add any other language. During the whole period of the programming of the system, the team consulted the approach often and regularly with the contracting authority, to fully meet the needs of the doctors.

---

[1]An abstraction in which software providing generic functionality can be changed by code additionally added by a programmer

2.1 Functional Requirements

A functional requirement defines a set of inputs, the behavior, and outputs of a system or its components. Functional requirements may include data manipulation and processing, data validation as well as implemented features such as searching, export, etc.

- Patient files

- Medical history records of patients

- Advanced Search according to given parameters

- Export of the records to sundry formats

- Verification of patients

- Managing users and roles

- Quicksearch in patients card index

- Validation of patients

- Versioning of database changes

- Hiding patients and their medical records instead of deleting

2.2 Non-functional Requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the system. Usually those are security, accessibility or reliability.

- Role-based access control

- Robustness

- Reliability

- Solution based on free technologies

- Security

- Responsive design[2]

---

[2]Design that provides optimal viewing experience on any kind of device

2.3   Used back end technologies

1. Java 7

   Java is the foundation for virtually every type of networked application and is the global standard for developing and delivering embedded and mobile applications, games, Web-based content, and enterprise software. Java is designed to enable development of portable, high-performance applications for the widest range of computing platforms possible. By making applications available across heterogeneous environments, businesses can provide more services and boost end-user productivity, communication, and collaboration—and dramatically reduce the cost of ownership of both enterprise and consumer applications.[5]

2. Apache Tomcat application server

   Apache Tomcat is an open source software implementation of the Java Servlet and JavaServer Pages technologies. Apache Tomcat is developed in an open and participatory environment and released under the Apache License version 2. Apache Tomcat powers numerous large-scale, mission-critical web applications across a diverse range of industries and organizations.[1]

3. MySQL Community Server

   The MySQL software delivers a very fast, multi-threaded, multi-user, and robust SQL database server. MySQL Server is intended for mission-critical, heavy-load production systems as well as for embedding into mass-deployed software.[7]

4. Spring Framework

   The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform. A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.[9]

5. Spring MVC

   Spring's Web MVC framework is designed around a DispatcherServlet that dispatches requests to handlers, with configurable handler map-
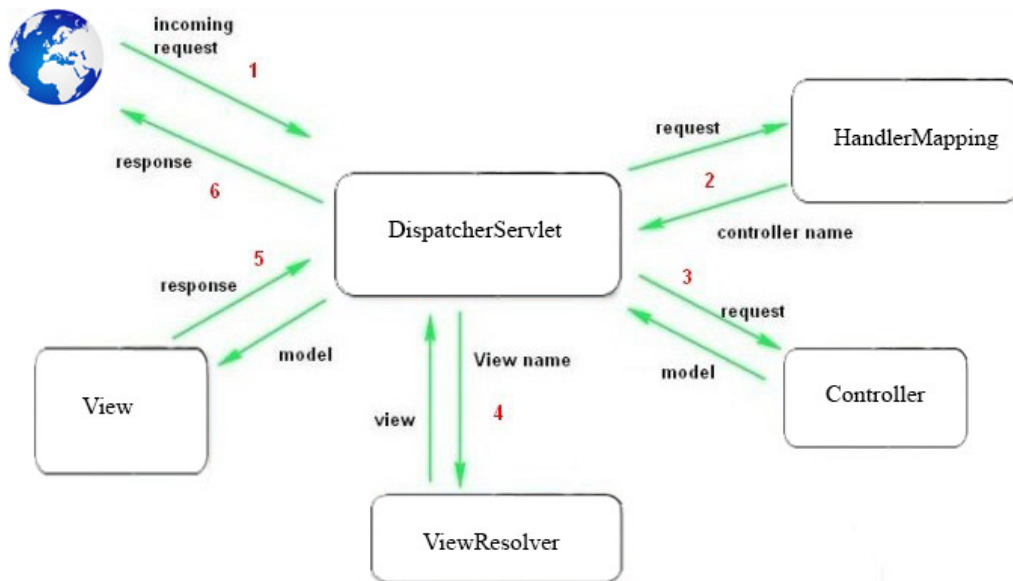
Figure 2.1: Spring MVC web-flow

pings, view resolution, locale and theme resolution as well as support for upload files. [11]

6. Spring Security

   Spring Security is a powerful and highly customizable authentication and access-control framework. It is the de-facto standard for securing Spring-based applications.[10]
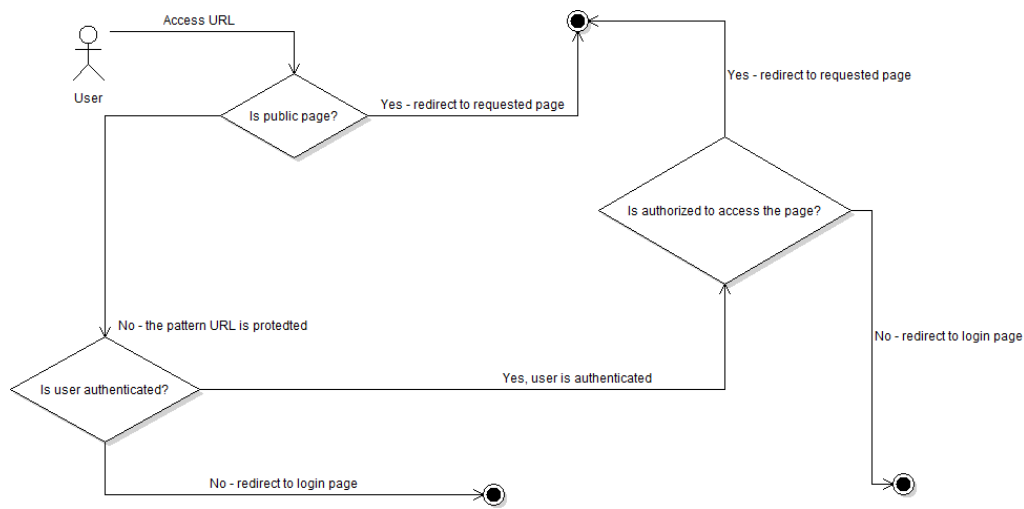
Figure 2.2: Authentication and Authorization flow

7. Hibernate

   Hibernate takes care of the mapping from Java classes to database tables, and from Java data types to SQL data types. In addition, it provides data query and retrieval facilities. It can significantly reduce development time otherwise spent with manual data handling in SQL and JDBC. However, unlike many other persistence solutions, Hibernate does not hide the power of SQL from you and guarantees that your investment in relational technology and knowledge is as valid as always.[8]

2.4   Used front end technologies

1. HTML5

   HTML5 contains powerful capabilities for Web-based applications with more powerful interaction, video support, graphics, more styling effects, and a full set of APIs[3]. HTML5 adapts to any device, whether desktop, mobile, tablet or television.[13]

2. JavaScript

   JavaScript (often shortened to JS) is a lightweight, interpreted, object-oriented language with first-class functions, most known as the script-

---

[3]A set of functions, classes or protocols of a library that can be used by a programmer

ing language for Web pages, but used in many non-browser environments as well such as node.js or Apache CouchDB. It is a prototype-based, multi-paradigm scripting language that is dynamic, and supports object-oriented, imperative, and functional programming styles.[4]

3. CSS

   Cascading Style Sheets (CSS) is a simple mechanism for adding style (e.g., fonts, colors, spacing) to Web documents.[14]

4. JSP

   JavaServer Pages (JSP) technology enables Web developers and designers to rapidly develop and easily maintain, information-rich, dynamic Web pages that leverage existing business systems. As part of the Java technology family, JSP technology enables rapid development of Web-based applications that are platform independent. JSP technology separates the user interface from content generation, enabling designers to change the overall page layout without altering the underlying dynamic content.[6]

5. Twitter Bootstrap

   Twitter Bootstrap is the most popular front-end framework for developing responsive, mobile first projects on the web. [12]. It is built with HTML, CSS and JavaScript. Thanks to this framework it was much easier to create a responsive front end with a pleasant and ergonomic look and feel.

6. jQuery

   jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.[3]

Goals

The goal of my bachelor's thesis was to create the reporting module for the GENEPI IS, that would meet the requirements of the health care professionals and researchers that will use it.

Since the GENEPI IS is accessed by various types of users, it was needed to adapt the reporting module to this fact. One of the requests to meet this end was that the data has to be able to be exported to more formats. Specifically, the formats pdf, docx, xls, csv and txt.

Also among the important requirements included was that it has to be able to modify the scope of the exported data based on user needs, save and load export configurations, automatic and optional data anonymization and user-friendly interface.

Of course there has to also be data security to prevent access to sensitive patient data to users without proper permission.

The main goals consequently were:

- Implement export to pdf, docx, xls, csv and txt format

- Implement export configuration

- Allow the user to save and load customized configuration

- Allow the user to anonymize exported data

- Anonymize exported data automatically if the user doesn't have sufficient privileges
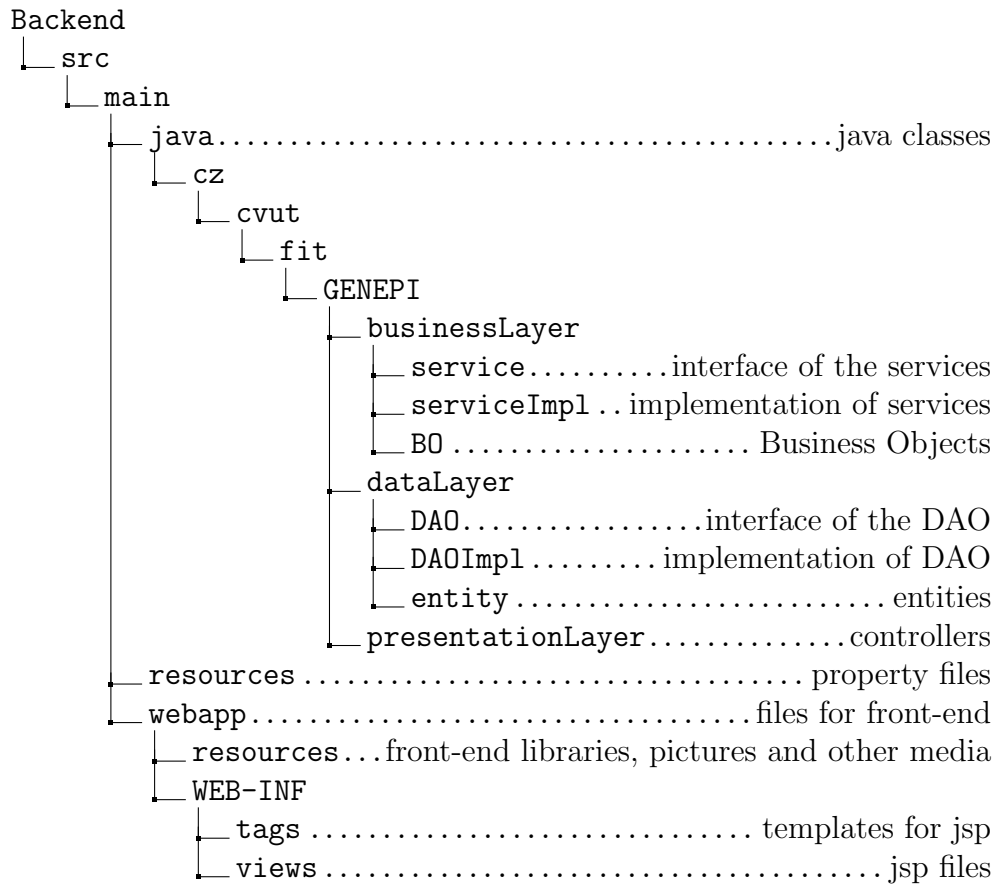
CHAPTER **4**

Analysis and design

The reporting module, as well as the whole information system, was designed given the requested robustness, accessibility, reliability and the cleanness of the source code. GENEPI IS has a three-tier architecture driven by a Spring MVC module, uses access according to the roles of the users via Spring Security module and thanks to the optimalized data layer using the Hibernate framework, it saves the computing resources and facilitates developers to understand and extend the source code.

Due to the fact that users who will work with the exported data have different levels of access and different requirements on the format of the exported data there was a need to make the module able to anonymize sensitive data as well as to export data to different formats. The contracting authority also requested that the user could choose which data he would want to export.

4.1 Design of the reporting module

Architecture of the reporting module doesn't differ from the architecture of the rest of the information system. Thanks to this I was able to guarantee the robustness and the security of this module. It has also provided me an easy and elegant way to access the other components in the system.

In this tree diagram you can see the parts of the IS that concern the reporting module:

```
Backend
└── src
    └── main
        ├── java..........................................java classes
        │   └── cz
        │       └── cvut
        │           └── fit
        │               └── GENEPI
        │                   ├── businessLayer
        │                   │   ├── service..........interface of the services
        │                   │   ├── serviceImpl..implementation of services
        │                   │   └── BO .................... Business Objects
        │                   ├── dataLayer
        │                   │   ├── DAO................interface of the DAO
        │                   │   ├── DAOImpl.........implementation of DAO
        │                   │   └── entity.........................entities
        │                   └── presentationLayer..............controllers
        ├── resources ................................ property files
        └── webapp...................................files for front-end
            ├── resources...front-end libraries, pictures and other media
            └── WEB-INF
                ├── tags .............................. templates for jsp
                └── views .......................................jsp files
```

4.1.1   Design of the back end part

The back end part of the reporting module was designed to follow the three-tier architecture of the rest of the system. Thus the classes that this module uses, are devided into three different packages. Every package contains classes that belong to the same tier. These packages are called Presentation layer, Business layer and Data layer.

4.1.1.1   Presentation layer

In the presentation layer there is a controller - a Spring MVC component that calls up a function according to the HTML request and its URL that is mapped to a particular function. This function executes simple operations such as verification of passed parameters and calls to the other layers. The results of the functions from the other layers may be saved to a org.springframework.ui.Model object and passed to the front end. This layer never executes more complicated actions as those should be done

within the Business layer, nor does access the database as this should be done within the Data layer. The functions of a controller usually return the name of a view - a .jsp file - that should be displayed to a user.

4.1.1.2 Business layer

There are three packages in this tier - concretely these are service, serviceImpl and BO.

Service contains interfaces whereas serviceImpl contains classes that implement those interfaces from the service package. It is analogical as in the Data tier which I will describe in greater detail in the next paragraph.

In serviceImpl classes there is usually implemented some more complicated logic that shouldn't be implemented within the Presentation layer, like adjusting of BOs, sorting of lists of objects under some conditions as well as many other actions.

The third package - BO - contains classes that transform the classes from the form in which the data come from the database - in the entity classes - to a form that is more appropriate for work within the Business tier.

4.1.1.3 Data layer

This layer contains three packages - DAO, DAOImpl and entity. In the DAO package there are interfaces of the data access objects, in DAOImpl are the classes that implement the interfaces from the DAO package and finally in the entity package, we can find the entity classes.

In DAOImpl classes you find the functions that are used to access the database and to get some data from it. There is a direct use of hibernate functions or queries in HQL.

In the entity classes is described how the structures of the tables in the database look. Every entity contains a property for every single column of the table. The type of this variable must match the type of the column of the table. If these conditions aren't satisfied the database query will fail and the AS will throw out org.hibernate.MappingException.

Every property has to have a getter and a setter as well. Thanks to anotations from javax.* and org.hibernate.validator.constrains.* it is possible to check the requirements for attributes of the input that should be saved to the database, i.e., minimal or maximal size, regexp pattern or if the input is blank or NULL.

A Hibernate annotation, @Column, is also mandatory as it describes the mapping of the property to the particular table in the database.

13

I would like to point out the generic GenericDAO and GenericDAOImpl classes that have been implemented. These are abstract classes that were implemented to ease the process of programming of the other DAO and DAOImpl classes.

In the GenericDAOImpl class there are implemented the most used methods that may be used by any service such as save, delete, getCount, getCountOfUnhidden, findByID and many other. The GenericDAO class is an interface of the GenericDAOImpl class as is usual in the rest of the project.

The other DAO and DAOImpl methods extend these generic classes so that it wasn't subsequently needed to implement these basic functions to every single class and the programmer had to programme only those functions that were unique for that particular class.

4.1.2 Design of the front end part

As well as the back end part, the front end had to follow the style and design of the rest of the system. If I haden't done so, the work with the reporting module could be confusing for users. On the basis of these facts, it was needed to implement a front end based on the same technologies as the rest of the GENEPI IS. Thanks to this approach the User Experience[4] from work with this module is positive and the user accepts the reporting module as an integral part of the IS.

---

[4]User's behaviors, attitudes, and emotions about using a system or service

CHAPTER 5

Realisation

In this chapter, I would like to describe how the export, its customization and security was implemented. I would also like to describe the inner logic used while performing these actions. In the end, it is also needed to familiarize the reader with the form of the front end and describe its sections individually.

The standard approach when performing an export is as follow:

1. The user chooses in the export view the properties that he wants to include in the exported file, the format, anonymization option and hits the export button

2. The PatientExportPOST method in the PatientController class is called up and according to the given parameters it calls up the method to export the data to the right format. It also checks if the data should be anonymized and if the user isn't authorized to see sensitive data then it sets the anonymize flag to true automatically. All given parameters with all of this information are passed to the called upon export function in the Bussiness tier.

3. In this step we've moved from the Presentation tier to the Business tier, where the methods from the Data layer are called to get patient data. Then one of the methods for export to a particular format is called. This is the part where the data is being written to the file itself. The program iterates through all the patients that the user wanted to export and gradually prints out their data. First the patient's header is printed out. This header contains information about the patient such as his age, gender, age at the onset of epilepsy and other

important data. It may also contain sensitive data such as his name, address or other contact information.

4. In the last step the user is redirected to an URL of the newly created file with exported data to download. This file persists on the server and can be repeatedly downloaded. The time that this file is accessable depends on the setting of the server CRON[5], as this should regulary delete old files in the directory dedicated for export files.

The goal was to make the whole process secure, reliable and easy to understand for any programmer reading the code, as well as for the user accessing the reporting module. Thanks to the chosen design and implementation approach these requirements were satisfied.

---

[5]A time-based job scheduler in Unix-like operating systems.
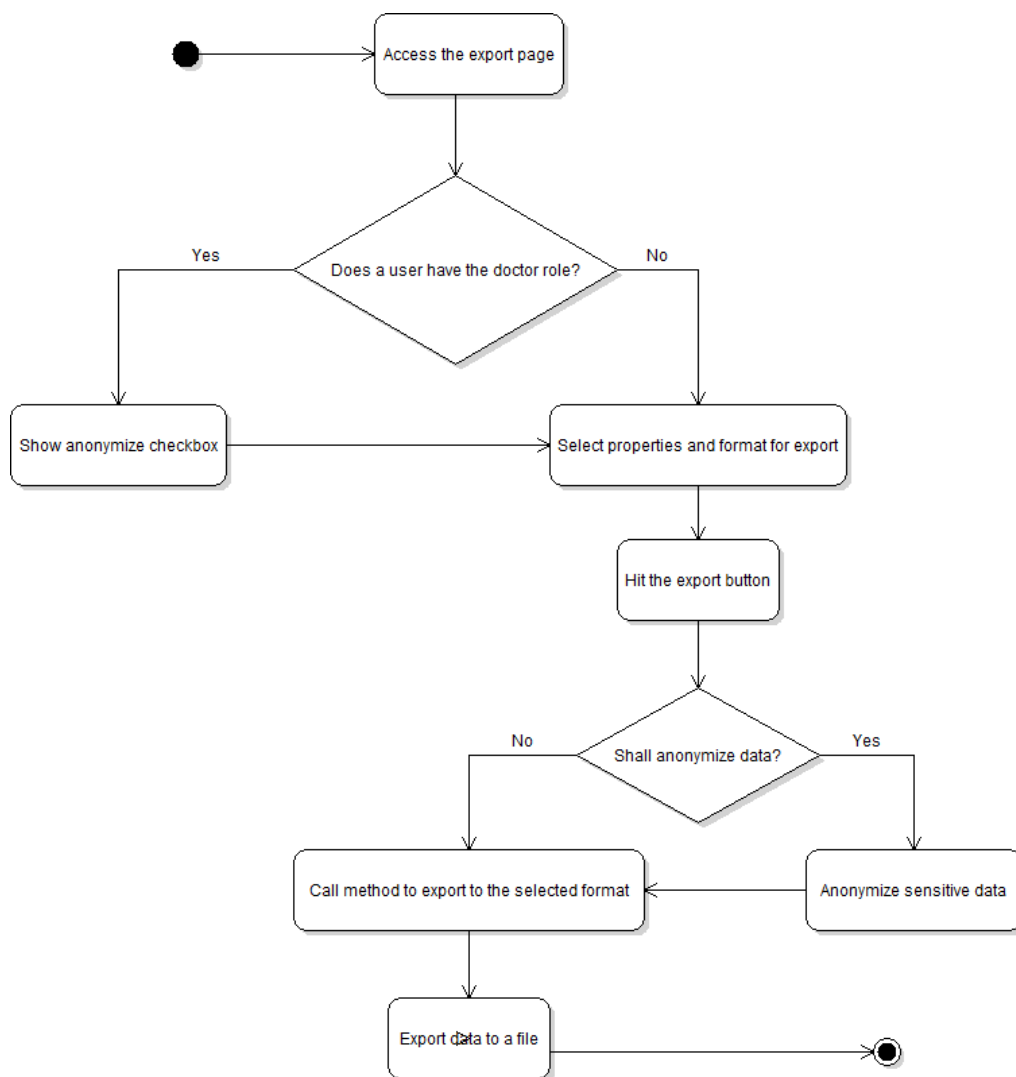
Figure 5.1: Export activity diagram

5.1 Customization of the export

There was a strong demand from the contracting authority to create a module that would let the user customize the report. Every patient can have fifteen cards filled in. This means arround 340 properties that can be stored for every single patient. Nevertheless not all of them are needed to be exported in certain situations. Therefore it was necessary to implement some solution that would allow a user to select only those cards or properties of those cards that should be exported.

Due to the high amount of those properties, it was quite challenging to create a solution that would meet the requirements and will be user-friendly at the same time. I've chosen a treeview on the front end and a special entity on the back end.

On the front end there was implemented a customized treeview, where every item has a checkbox. Patient's cards are represented by the nodes, whereas properties of these cards can be represented by the leaves of this tree or nodes as well. They may be nodes if they denote some category that can hold some properties that couldn't exist without their parent category. By checking and unchecking these items, the user can choose, what all should be involved in the exported data. When unchecking the node all leaves of this node are aslo deselected. On the other hand, by selecting even a single leaf from an unselected branch, the parent node is selected as well. Here I attach a simple example of the treeview I was writing about:

```
  Anamnesis.......................................anamnesis card
  ├──Anamnesis property
  ├──Another anamnesis property
├──Seizures..........................................seizure card
  ├──Seizure property
  ├──Another seizure property
├── ...
├── ...
├──Neuropsychology.............................neuropsychology card
  ├──Neuropsychology inner node
     ├──Neuropsychology property
     ├──Another Neuropsychology property
  ├──Another Neuropsycholgy inner node
     ├──Neuropsychology property
  ├──Neuropsychology property
├── ...
├── ...
├──Outcome ..........................................outcome card
```

After confirming the export form, the state of the treeview is saved to an instance of an entity called ExportParams. This entity has a boolean property for every single value from the treeview in the export form. Thanks to this I am able to determine which properties the user wanted to export and which he didn't. It gives me another advantage - I can save the current setting of the treeview for a later usage, but I will write about this feature later.

5.2 Export to the particular formats

During the programming of the classes that procure the logic of the reporting, I was trying to use the fact that there already exist java libraries that can export data to docx, pdf, xls and csv formats. I also avoided the duplication of the code by transforming data from one format to another. Thanks to these measures, it is much easier to make changes in the code of the classes that handle export itself and it also eases the understanding of the code for anybody reading it.

5.2.1 txt

Export to the text format is realized by components from java.io.* package. Specifically java.io.FileOutputStream, java.io.OutputStreamWriter and finally java.io.BufferedWriter. Output is encoded to the UTF-8 format. Every property is printed out to a new line, sections are delimited by dash lines, star lines or empty lines.

5.2.2 docx

Export to the docx format was not as easy as to the txt, so I decided to look up suitable libraries, compare them and use the one that would best suit my needs. After researching the possible solutions I ended up with two libraries - Apache POI and docx4j.

I've chosen to use docx4j because it provided me more functionalities and clearer API then Apachi POI could. The main problem I had with Apachi POI was the formating of the cells when I wanted to format data to the tables. I could hardly adjust the size of the cells and redefine my own styles of headers and text. Docx4j provided me a really easy and elegant way to solve these issues.

5.2.3 xls

As well as for the docx format, I used the fact that there already were programmed libraries for export to xls. I've chosen to use Apache POI library, as this library provides me eveything I need to export data to an xls file, such as merged cells, creation of new lists and changing the background color of the cells.

5.2.4 pdf

There are of course java libraries that provide you some way to export data to pdf format as well, nevertheless I've chosen a different approach. While I already had implemented the export to docx, I've chosen not to implement export to pdf, but instead to create a file with data exported to docx and using the classes from apache.poi.xwpf.converter.* package to convert this file to pdf. Thanks to this approach I didn't duplicate the code because the only logic needed to format and print out the data for export was already implemented in the export to docx. After transforming docx file to pdf, the original docx file is deleted and only the newly created pdf file persists.

5.2.5 csv

Before I started to implement the export to csv format, I was also thinking whether I should implement the whole logic of export to csv, but decided to proceed similarly as in the case of implementation of the export to pdf. Nevertheless, in this case I don't use a file with data exported to docx, but a file created by export to xls, as this format can be easily transformed to csv and vice versa. During the export to csv I create at first an xls file with exported data and then walk through this file and print out the values to the file via the classes from java.io.* package. These are the same classes that have been used in the export to txt. Based on the requirements of the contracting authority commas, as a standard delimiter in csv format, has been replaced with semicolons. The main reason of this change was the fact that every card that a patient may have contains a comment field, which is text and which may contain commas. As the semicolon is a much less used character in sentences, it was decided to use it as a delimiter. Otherwise the report could look confusing for the user.

5.3 Security and anonymization

In GENEPI IS there are 5 main levels of access, according to the visibility of the patient's data.

1. Users

   The User role is a basic role that every new user gets from the system by default. This role allows you to acces the homepage and your user profile. Without any further role, the user is not allowed to perform any further actions.

2. Administrators

   Administrators don't have any access to the patient's data. They are not even able to access the URL to view or export this data. They're the only users who may create new users and edit users' roles. They may also edit a users' details and change passwords.

3. Researchers

   Researchers have limited access to patients' data - they are not allowed to see sensitive data. Nevertheless they are still able to see lists of users, with limited options to use advanced search and export anonymized data. Researchers are not allowed to modify patients' data.

4. Doctors

   Doctors have full access to patients' data, they are allowed to add new
   patients and modify their data. Nevertheless, after every change the
   patient is set as unverified. This means that those changes weren't
   checked by an authorized person, can't be with certainty trusted and
   shouldn't be included in any research. Doctors by default don't have
   authority to verify patients whose data were changed and are set as
   unverified.

5. Superdoctors

   Users with the role of Superdoctor have the same privileges as users
   with the role of Doctor. Moreover they have the right to verify un-
   verified patients.

Because of this fact it was needed to implement some way to anonymize
the exported data. To the reporting module I implemented two types of
anonymization - optional and manadatory.

Optional anonymization is accessible to doctors and superdoctors only.
Before they perform the export, they can decide if they want to anonymize
the exported data. This could be useful, for example, in situations where
the doctor needs to provide the data to somebody who normally doesn't
have access to the system and at the same time shouldn't see the sensitive
informations. On the other hand the doctor still has the possibility to
export unanonymized data, which might be useful for medical purposes.

Mandatory anonymization is done automatically, when somebody who
doesn't have a sufficient access level wants to export data. This feature will
prevent anybody who is unauthorized from seeing the sensitive data of the
patients. Anonymized data will be mostly used by researchers who make
analysis upon the data and don't need to know the names and addresses of
patients because they can distinguish them according to their IDs, which is
a mandatory property printed out for every patient in every export.

5.4 Custom configurations

Customization of the configuration of exported data was one of the key
features of this module. Due to the fact that there are several types of
users who work with this IS it's necessary to take account of how the ways
they need to use it differs. Researchers usually do some analysis above
some part of the data from the IS, whereas doctors usually need some more
complex report about a patient's state of health and the treatment process.

It is obvious, that setting these configurations will be required regulary, as it is not expected that users would need to create a unique configuration for every single export he'd like to perform. Because of this, it was decided that users should have an option to save their adjusted configuration for a later use. The reporting module for GENEPI IS distinguishes two types of saved configurations.

The first is the configuration saved by a user with the Superdoctor role. This configuration may be set as generic. That means that the user chose to allow his configuration to propagate through the system so other users may see and use it.

The other type is the user's private configurations. Every user with a sufficient access level to access the export view is allowed to create and use them. These are configurations that were created by a user himself and are available only to him. The other users are not able to use these configurations, nor to see them.

Deletion of configurations is possible as well. Private configurations can be deleted by a user who can see them in his list of private configurations and not by anybody else.

On the other hand generic configurations are visible to all users in the system, therefore it is necessary to determine if a user who wants to delete some generic configuration has even the right to perform this action. To delete a generic configuration the user has to own a Superdoctor role. Then he's allowed to delete any generic configuration created by any other user. If the user doesn't have a Superdoctor role, then the delete button for generic configurations is not even displayed to this user in the export view. In the following pictures you can see the activity diagram of saving and then deleting of a configuration from the point of view of the user.
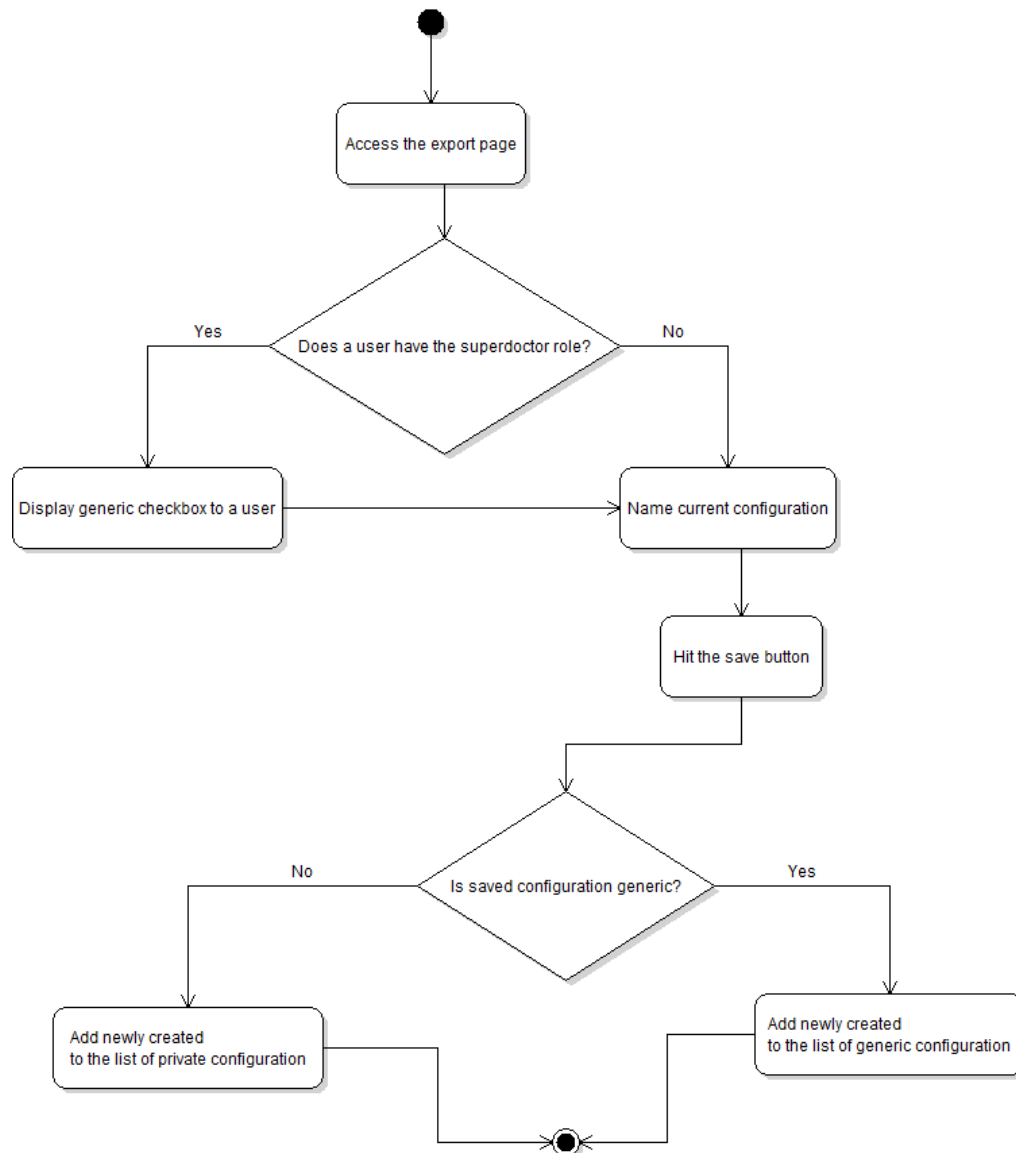
Figure 5.2: Create configuration activity diagram
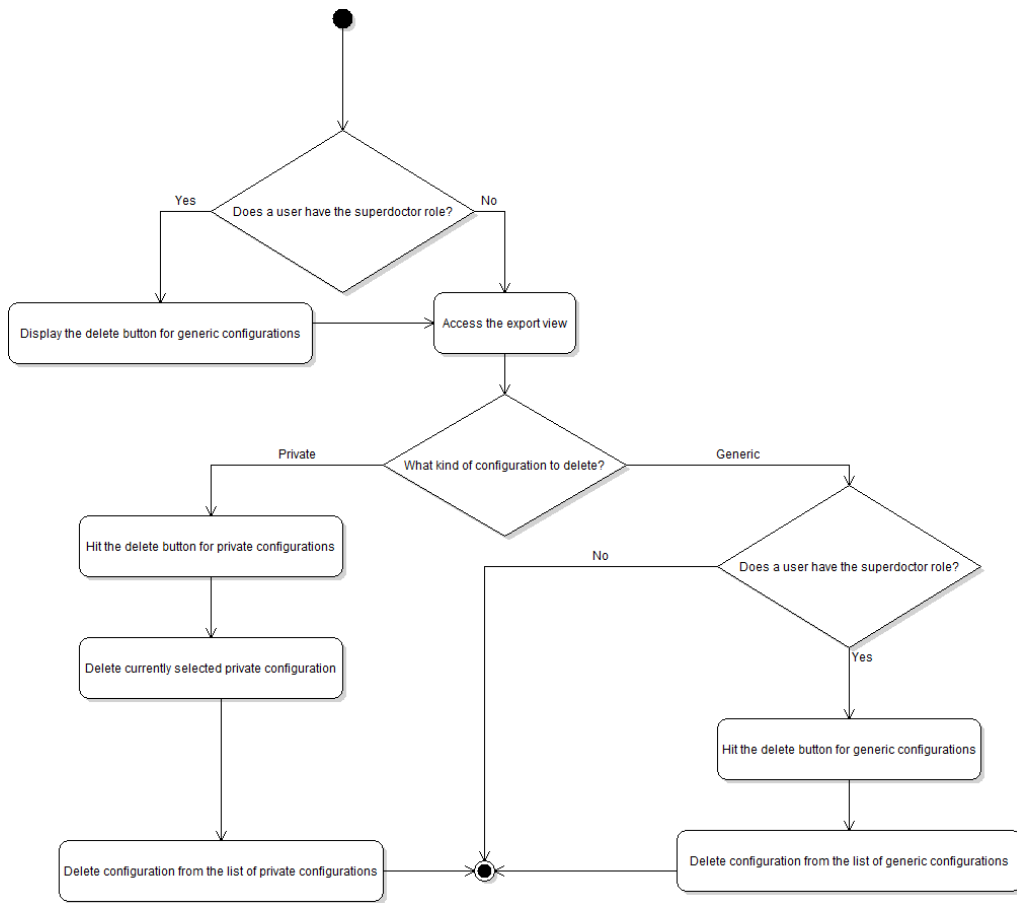
Figure 5.3: Delete configuration activity diagram

5.5 Graphical user interface of the reporting module

In the following pictures I'll explain all the components used to control the export view. During the phase of design and implementation of the view I had to take into account the fact that users of this module need a user-friendly interface that would provide them all of the tools necessary customize exports.

In the first picture, you can see the interface for loading and deleting custom configurations. In the first row there is a combobox with generic custom configurations which the user may load or even delete, if he's the creator of this configuration.

In the next picture there is displayed a section for export itself. First

Figure 5.4: Loading and deleting of a custom configuration

a list of IDs of the patients that are prepared for an export. The user may access the patient's overview by clicking his ID. Next it's necessary to choose the format for export via several radiobuttons. In the case that the user chooses as a format docx or pdf, a checkbox appears and allows him to choose if exported data should be formated into tables or as a text. If the user has a Doctor or Superdoctor role assigned, then he's able to see a combobox that allows him to anonymize exported data.

In the last picture you can see the interface for saving custom configurations. There is a textfield for naming the configuration and if the user has a Superdoctor role, then he may check the checkbox to make it generic.

Export

Patients to export

1906 1917 1935 1939 2002 2004 2015 2024 2089 2093 2095 2105 2122 2126 2188 2189 2192 2212 2235
2248 2293 2302 2303 2320 2334 2356 2370 2404 2412 2455 2456 2460 2497 2503 2561 2696 2735 2746
2794 2831 2939 2979 3038 3067 3112 3126 3234 3241 3242 3266 3271 3279 3337 3338 3380 3381 3382
3392 3398 3399 3401 3405 3406 3407 3409 3410 3411 3560 3567 3581 3589 3590 3592 3593 3594 3595
3596 3597 3600 3601 3602 3609 3629 3630 3631 3633 3634 3636 3637 3638 3639 3643 3644 3645 3646
3647 3648 3654 3651 3652 3653 3655

Settings

Choose format   ◉ pdf ○ xlsx ○ docx ○ txt ○ csv

☐ Export as a table

☑ Anonymize

[Export]

Parameters

☑ Check/Uncheck all
 + ☑ Anamnesis
 + ☑ Seizures
 + ☑ Pharmacotherapy
 + ☑ Neurological finding
 + ☑ Neuropsychology
 + ☑ Neuropsychology - old
 + ☑ Diagnostic test - Scalp EEG
 + ☑ Diagnostic tests - Neuroimaging test
 − ☐ Invasive Test ECoG
   − ☐ ECoG done
     ● ☐ ECoG coverage
     ● ☐ ECoG patterns
     ● ☐ ECoG After resection
     ● ☐ Comment
 − ☑ Invasive Test - iEEG
   − ☑ EEG done
     ● ☑ Intracranial electrodes
     ● ☑ localization intracranial electrodes
     ● ☑ EEG Slow
     ● ☐ Interictal EEG spikes
     ● ☐ Localization invasive EEG interictal spikes
     ● ☑ Invasive EEG status epilepticus
     ● ☑ Ictal EEG patterns
     ● ☐ Localization ictal EEG pattern
     ● ☑ Comment
 + ☑ Invasive test - Cortical mapping
 + ☑ Operation
 + ☑ Histology
 + ☑ Complication
 + ☑ Outcome

Figure 5.5: Selection of format and customization of an export

Figure 5.6: Saving a custom configuration

CHAPTER 6

Testing and Examples

Testing of the reporting module as well as the GENEPI IS itself was performed during the development by the developers themselves and by a member of the team who held the position of a tester. After key functionalities were implemented, Ing. Ježdík Ph.D obtained access to the system and tested it with normal use cases. He of course had many requests for changes that would improve the application or found bugs that requiered fixing. Currently the GENEPI IS with reporting module should be deployed on produciton servers in the Faculty Hospital Motol and will fully replace the existing application. As in most of the cases from the commercial sector, it is expected, that users will also make some requests for changes. But those are expected to be minor change requests, mostly on the front end and should be implemented within a few MD.

In the following screenshots the reader can see a few examples of how the export files created by the reporting module look. I also enclose a simple example of an analysis made on the basis of data from GENEPI IS, that were exported into xls. All these examples spring from real data, currently saved in the IS, but they were anonymized of course. You may find these files enclosed on the CD as well.

Figure 6.1: Screenshot of an export of all patients to docx

Figure 6.2: Screenshot of an export of all patients to docx

Figure 6.3: Screenshot of an export of all female patients to txt

Figure 6.4: Screenshot of an export of all patients older then 5 years to xls

	K	L	M	N	O	P	Q	R	S	T
1		Neuropsychology		Neuropsychology		Diagnostic test - Scalp EEG		Diagnostic test - Scalp EEG		Invasive Test ECoG
2			Neuropsychology - old from date	05/03/1999			Diagnostic tests - Neuroimaging tests from date	03/02/2010	Invasive Test ECoG from date	03/02/20
3			Neuropsychological examination	Yes			MRI done	Done	ECoG done	Not dor
4			Intelligence level	Average inteligence			MRI finding	Focal		
5			Specific learning	No			MRI description	hypothalamický hamartom při lat. stěně III. komory vel. cca 10x10x10 mm, bez progrese		
6			Developmental language disorders	No			FDG PET	Neprovedeno		
7			ADHD syndrom	No			Description of PET hypometabolism	Not available		
8			Comment	No comments			Icalt SPECT	Neprovedeno		
9							Description of SPECT hypoperfuse	Not available		
							Icalt SPECT	Neprovedeno		

Patient ID 2095 | Patient ID 2105 | Patient ID 2122 | Patient ID 2126 | **Patient ID 2188** | Patient ID 2189 ...

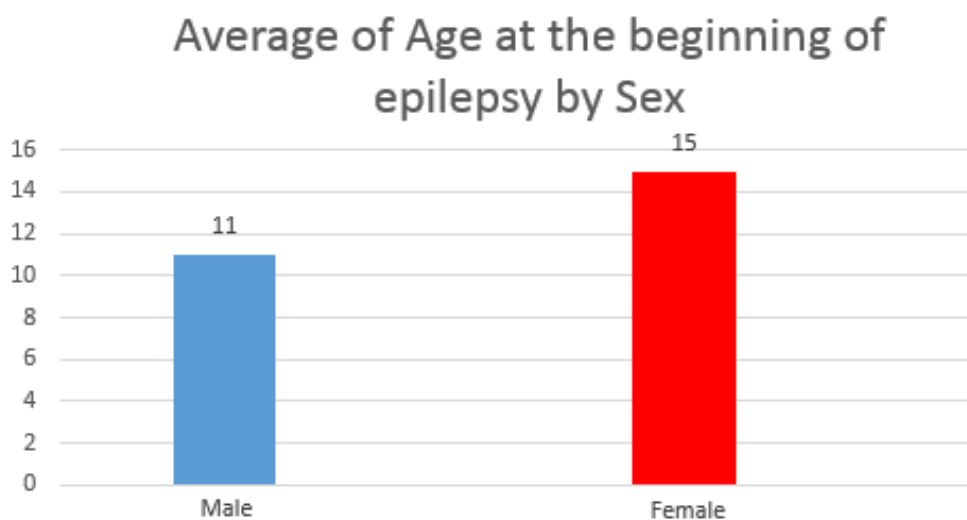Figure 6.5: Screenshot of an export of all patients older then 5 years to xls

Figure 6.6: Example of an analysis from exported data

Conclusion

I've created a tool that greatly extends the possibilities that GENEPI IS could provide to doctors and researchers who use it. Without the reporting module GENEPI IS would be only a tool to store and look up informations about patients. But now it becomes a tool, that can provide to users structured and customized reports. Those reports may be used as a source of important information about the patient's state of health and about the process of his treatment through surgery as well as a source of data for analysis and research.

Thanks to the possibility to customize exports the doctors can select to export only that data which is important for a given medical intervention. As it is expected that the set of requested information to given interventions will not change frequently, the doctors can also use the feature of the module to configure the export once, then save the configuration and then just load it anytime they would need it, without any need of further configuration.

Another possibility of using the reporting module are analyses made by researchers. Researchers shouldn't usually be allowed to see sensitive data about patients. Thus the feature of automatical anonymazation of the patient's data comes in handy. Also a possibility to export an unlimited number of patients during one export to one file is useful, because researchers need a larger amount of data to make any analysis. Exporting every single patient to separate files one by one, would be a really user-unfriendly solution. It is also expected, that researchers will use different formats than doctors. Whereas doctors are expected to use mostly docx or pdf format, researchers will probably use xls, csv or txt. These formats can be easily parsed and processed to other software for further analysis.

For all these reasons it is now possible to execute export to 14 sundry formats. Those are 5 basic formats, 5 more in anonymized variant. Docx

and pdf export can be also formated to tables. With the anonymized variant we get 4 more formats. This solution should be sufficient to meet the needs of any user of GENEPI IS.

To sum up, all goals stated in the beginning of this thesis were satisfied. Moreover, I've added some features that may ease the work with the reporting module, such as generic configurations or more types of formatting when exporting to docx and pdf.

I must say I'm really satisfied with the software I've created on the basis of the requests of the doctors from the Faculty Hospital Motol in Prague. I belive that it will serve doctors and researchers well and make their work and research much easier. This all should lead to an improvement of the treatment of patients with epilepsy.

I'm proud that result of my bachelor's thesis will be deployed in the biggest hospital in the Czech Republic and will be used there to help real patients. I also believe that my work will have its share on helping all mankind to understand such an insidious disease such an epilepsy is and maybe even to save somebody's life.

Bibliography

[1] Apache Software Foundation: *Apache Tomcat Home.* [cit. 2014-05-04]. Available at WWW: `http://tomcat.apache.org`

[2] Jenkins CI Community: *About Jenkins CI.* [cit. 2014-05-04]. Available at WWW: `http://jenkins-ci.org/content/about-jenkins-ci`

[3] jQuery team: *What is jQuery.* [cit. 2014-05-04]. Available at WWW: `https://jquery.com`

[4] Mozilla Foundation: *JavaScript.* [viewed 2014-05-04]. Available at WWW: `https://developer.mozilla.org/en-US/docs/Web/JavaScript`

[5] Oracle corporation: *About Java.* [viewed 2014-05-04]. Available at WWW: `http://www.java.com/en/about`

[6] Oracle corporation: *JavaServer Pages Technology.* [viewed 2014-05-04]. Available at WWW: `http://www.oracle.com/technetwork/java/javaee/jsp/index.html`

[7] Oracle corporation: *MySQL 5.5 Reference Manual.* [viewed 2014-05-04]. Available at WWW: `http://dev.mysql.com/doc/refman/5.5/en/introduction.html`

[8] Red Hat: *Hibernate Community Documentation.* [viewed 2014-05-04]. Available at WWW: `http://docs.jboss.org/hibernate/orm/4.2/quickstart/en-US/html/pr01.html`

[9] SpringSource: *Spring Framework Introduction.* [viewed 2014-05-04]. Available at WWW: `http://projects.spring.io/spring-framework`

[10] SpringSource: *Spring Security Home.* [viewed 2014-05-04]. Available at WWW: `http://projects.spring.io/spring-security`

[11] SpringSource: *Web MVC framework Introduction.* [viewed 2014-05-04]. Available at WWW: `http://docs.spring.io/spring-framework/docs/2.0.x/reference/mvc.html`

[12] Twitter, Inc.: *Twitter Bootstrap Home.* [viewed 2014-05-04]. Available at WWW: `http://getbootstrap.com`

[13] WWW Consorcium: *w3 wiki FAQs.* [viewed 2014-05-04]. Available at WWW: `http://www.w3.org/html/wiki/FAQs`

[14] WWW Consorcium: *What is CSS.* [viewed 2014-05-04]. Available at WWW: `http://www.w3.org/Style/CSS`

Acronyms

API Application Programming Interface

AS Application Server

BO Business Object

CI Continuous Integration

DAO Data Access Object

GUI Graphical User Interface

HQL Hibernate Query Language

IS Information System

JDBC Java Database Connectivity

JSP JavaServer Page

MD Manday

MVC Model View Controller

XML Extensible Markup Language

Contents of enclosed CD

To my bachelor's thesis I enclose a CD which contains all the important data, needed to deploy GENEPI IS. It also contains source codes of GE-NEPI IS and the libraries that it uses. In the installPackage directory is also in the migrate folder an enclosed a model for MySQL Workbench, where the reader may see the structure of the database. In the same folder you can also find a directory with migration scripts. These are important when migrating from the information system that was previously used in the Faculty Hospital Motol to GENEPI IS. Migration scripts allow you to copy data from the previously used database to the database that is being used by the GENEPI IS. Thanks to this, the reader should be able to administrate or even modify GENEPI IS without any larger problems.

The next important directory is the directory with samples of exported data. This folder contains 14 files with all possible formats of exported data - pdf, docx, xls, csv, txt - all of these formates in both anonymized and unanonymized variants (sensitive data were changed) and both printed as a plain text and formated to a table for pdf and docx as well.

The reader will also find a useful directory that contains deployment packages of the results of the previously mentioned projects from the subjects BI-SP1 and BI-SP2. The documentation to these doesn't differ from the documentation for the GENEPI IS.

I would also recommend you become familiar with the content of the directory documentation, where you will find user and admin documentation for GENEPI IS.

Additional content of the enclosed CD is the text directory, that contains this thesis in a pdf format.

```
documentation........the directory of the GENEPI IS documentation
installPackage.the directory of install package of the current version
previousVersions..the direcotry of the results of BI-SP1 and BI-SP2
sampleData......the directory that contains samples of exported data
source the directory of source codes of the GENEPI IS and this thesis
    impl..............the directory of source codes of the GENEPI IS
        images..............the directory of images used in the thesis
        thesis.ps.......................the thesis text in PS format
        Backend......................the directory of a maven project
            src..............the directory of GENEPI IS source codes
                main
                    java
                        cz
                            cvut
                                fit
                                    GENEPI.......the directory of java classes
                    resources............the directory configuration files
                    webapp
                        resources..the directory of librarires for front end
                            WEB-IF
                                spring...the directory for spring context file
                                tags.the directory of templates for jsp pages
                                views.............the directory of jsp pages
                test..........the directory of source codes of unit tests
    thesis.................the directory of source codes of this thesis
text.......................................the thesis text directory
    BachelorsThesis.pdf..............the thesis text in PDF format
readme.txt....................the file with CD contents description
```