Remarks

4: (Nie jestem rodowitym użytkownikiem angielszczyzny, zatem moje uwagi językowe proszę brać z rezerwą) has
4: for citizens is
5: Poniższy tekst jest do przeniesienia do któregoś z następnych rozdziałów.
7: Trochę inaczej się cytuje – poprawiłem w pliku `thesis.bib`. Wpisy bibliograficzne w BibTEXu są zwykle dostępne w internecie.
11: Czy użytkownik, który się jeszcze nie zalogował «citizen» może głosować?
14: It is
14: it is
14: spacja przed cytowaniem

SILESIAN UNIVERSITY OF TECHNOLOGY

FACULTY OF AUTOMATIC CONTROL, ELECTRONICS, AND COMPUTER SCIENCE

Engineer thesis

Central online voting system

author: Wojciech Drzewiecki

supervisor: Krzysztof Simiński, PhD DSc

Gliwice, January 2020

załącznik nr 2 do zarz. nr 97/08/09

Oświadczenie

Wyrażam zgodę / Nie wyrażam zgody* na udostępnienie mojej pracy dyplomowej / rozprawy doktorskiej*.

Gliwice, dnia 27 stycznia 2020

.....
(podpis)

.....
(poświadczenie wiarygodności
podpisu przez Dziekanat)

* podkreślić właściwe

Oświadczenie promotora

Oświadczam, że praca „Central online voting system” spełnia wymagania formalne pracy dyplomowej inżynierskiej.

Gliwice, dnia 27 stycznia 2020

.....

(podpis promotora)

Contents

Chapter 1

Introduction

Chapter 2

Online voting systems

2.1 Introduction

Ever since internet was introduced in our lives, online election voting was a contentious issue.

First of all, online voting would be convenient—right now voting is associated with going to a nearby ward, waiting in a queue, complicated procedure of receiving a ballot and filling a ballot. Moreover, waiting for results could be decreased—now we have to wait for all the votes to be counted, protocols sent to commissions, summed up, and determination of the winners. All this can take over few dozen hours. Online voting would decrease that time hugely, at the same time being much more precise than counting by a person.

On the other hand, there is significant trust issue. There will always be a person which have access to data, and with such access that person could change elections results in a matter of seconds. Following this lead, even if nobody has access to data, we will never be sure that vote we cast is not changed somewhere in the logic of the application. We can eliminate that by developing open source application, which can be verified that it does not change something in the logic. This approach would create many more problems, of which two are: firstly, open source application can be investigated by everyone, and with given source it is much easier to find a hole in the logic from which we can benefit. Secondly, even if we see view of that open source application, we can never be sure that logic

behind it is not swapped.

You can see a pattern here—for every argument there is a counter argument, which generates another problems. There is no right answer in this debate.

2.2 Ideal voting system

So what would be features of the perfect voting system? Let's investigate it along with flow of online voting.

Before election, system administrator has to enter committees and candidates data into a system, register them for polls predicted for certain date, which he have [(Nie jestem rodowitym użytkownikiem angielszczyzny, zatem moje uwagi językowe proszę brać z rezerwą) has] entered at the beginning. Although he could handle registering candidates to certain committees, he cannot handle scope of managing whole committee during election. It's best if administrator granted some chosen user privileges to manage certain committee inside the system—let's call him committee administrator. It's useful for example in case of allocation seats via D'Hondt method—it's committee itself that wants to decide on order in closed list, there should be no third party involved.

First, in order to cast a vote through internet, a citizen would have to register. What online voting system wants to achieve is for citizens [for citizens is] to be able to cast a vote without leaving home. It makes sense that citizens would register also through the internet—for example in case of some injury, if someone could go out of home, they could also vote in a ward. It creates a problem—how do we know that person on the other side of the screen is who he says he is. Solution is two step authentication process—citizen registers with citizen id (for example PESEL in Poland), he is sent a text message or an email, but also receives registered letter of registration, both with activation codes, that only together allows him to register properly. Registered letter allows to verify citizen willing to vote through internet without need of leaving the house. Because email can be generated without interference of any third party person, code sent this way secures citizen from someone taking advantage of code in letter. Those two ways combined secure citizens account from being taken advantage of.

Let's say a citizen registered successfully. On election day, he should be able to

vote on exactly the same polls as he would personally, in ward. Citizen should be able to cast a vote only once, and his vote should be detached from his account—in no way should anybody be able to tell which vote belongs to which citizen. Although registered for online voting, a citizen should still be able to vote personally in ward. However, once he voted in any of two ways, the second one should be automatically and immediately blocked. This requires for the system to work also in wards. Once user votes online or receives a ballot from commission in a ward, he should be immediately blocked for second type of voting.

Citizen easily and securely voted either through internet or personally. Now his journey ends, he can go back to living his life. What's left to do is to calculate results of election. However, system also has to take into account all votes casted in wards. Results can be calculated if and only if all wards have entered their protocols into the system. To do so, a member of a commission should be assigned with access to send the protocol to the system—let's call him a ward administrator. Once that member is a registered user, he should be able to be assigned as particular ward admin by the system administrator. Then, he should be able to file in the protocol, once the election is closed for voting.

After closing the election and collecting protocols from all wards, results of the election can be calculated. This information can be public and delivered to public on the same site that they voted on, as well as on official government electoral commission website.

2.3 Problem statement

[Poniższy tekst jest do przeniesienia do któregoś z następnych rozdziałów.]

2.3.1 Overview

Let's put above functionalities into more technical perspective. In order to secure application from unauthorized access, it is best if we divide application into four main cockpits:

- Voter cockpit

- Administrator cockpit
- Ward administrator cockpit
- Committee administrator cockpit

Each cockpit will have certain role that is necessary to even display available actions. In order to have access to anything, we have to be authenticated (active, successfully registered) user, with assigned role that gives us permission for certain cockpits. A guest—not registered user—is only available to register or login.

2.3.2 Voter cockpit

User with role of a voter should be able to vote for election, and see election results, after it's closed. In case of voting, user must be able to vote on a poll only once, without any exception.

2.3.3 Administrator cockpit

User with role of an administrator should be able to create essential data—wards, polls, committees etc. as well as granting certain roles to certain users. Administrator should not be able to interfere in ward or committee administration, other than choosing its administrator. Moreover, administrator should be the one to trigger polls results calculation.

2.3.4 Ward administrator cockpit

User with role of ward administrator should be able to manage only his assigned ward. Under no circumstances should he be able to manage other wards. In ward administrator cockpit he should be able to enter protocol from ward for each poll during an election.

2.3.5 Committee administrator cockpit

User with role of committee administrator should be able to manage only his assigned committee. Under no circumstances should he be able to manage other

committees. In committee administrator cockpit he should be able to choose order of committee candidates in closed list.

2.4 Overview of similar solutions

2.4.1 Electronic voting in Estonia

In Estonia, voting is based on electronic citizen ID. It's mandatory and sufficient national identity document, which allows for secure remote authentication. To cast a vote, an Estonian needs a computer connected to the internet and equipped with card reader. They can authenticate using digital certificate included in their citizen card, and cast a vote on the internet [3]. [Trochę inaczej się cytuje – poprawiłem w pliku `thesis.bib`. Wpisy bibliograficzne w BibTEXu są zwykle dostępne w internecie.]

This solution is problematic for several reasons:

- It's impossible to be easily implemented in countries without electronic citizen ID. Providing such documents for most of society of the country is a long time process, measured in years rather than months.
- A card reader capable of reading such a card is not a common device—a citizen has to go somewhere with a card reader, if he wants to cast a vote.
- Electronic citizen ID is not an intellectual knowledge like password or token. It's a physical device that can be easily stolen—and of course blocked in some department—but it may be too late and someone may have already taken advantage from owning someone else's citizen ID.

Chapter 3

Requirements and tools

3.1 Functional requirements

- Citizen of a country that holds an online election in the application should be able to register.
- Application should allow registered citizens to vote on candidates of their preference.
- Vote of a citizen should be taken into account when calculating results of an election.
- Citizen should be able to cast only one vote per election.
- Vote of a citizen should not be trackable—no person should be able to tell which citizen voted on which candidate.
- Person designated as a ward administrator—and only that person—should be able to send ward protocol to the system.
- Protocols sent from wards should be taken into account when calculating results of an election.
- Person designated as a commission administrator—and only that person—should be able to enter commission's closed list candidates order to the system.

- Administrator of the application should be able to perform steps that lead to creating an election identical to one provided to him.

- Administrator should be able to trigger calculation of the results of an election.

- Election results should not be calculated with some of wards' protocols missing.

- Administrator should be able to grant privileges of ward and commission administrators to designated people separately.

- Administrator should not be able to perform actions in replacement of ward or committee administrator.

3.2 Nonfunctional requirements

- User interface should be understandable to English speaking individual.

- Application should be implemented in Java, with help of Spring Boot and related frameworks for backend, and Thymeleaf template engine for frontend.

- Application should store all necessary data in MySQL database.

- Application should hash all sensitive data stored in database.

- Application should offer all available functionalities on Google Chrome version 79.

3.3 Use cases

3.3.1 Citizen use case UML

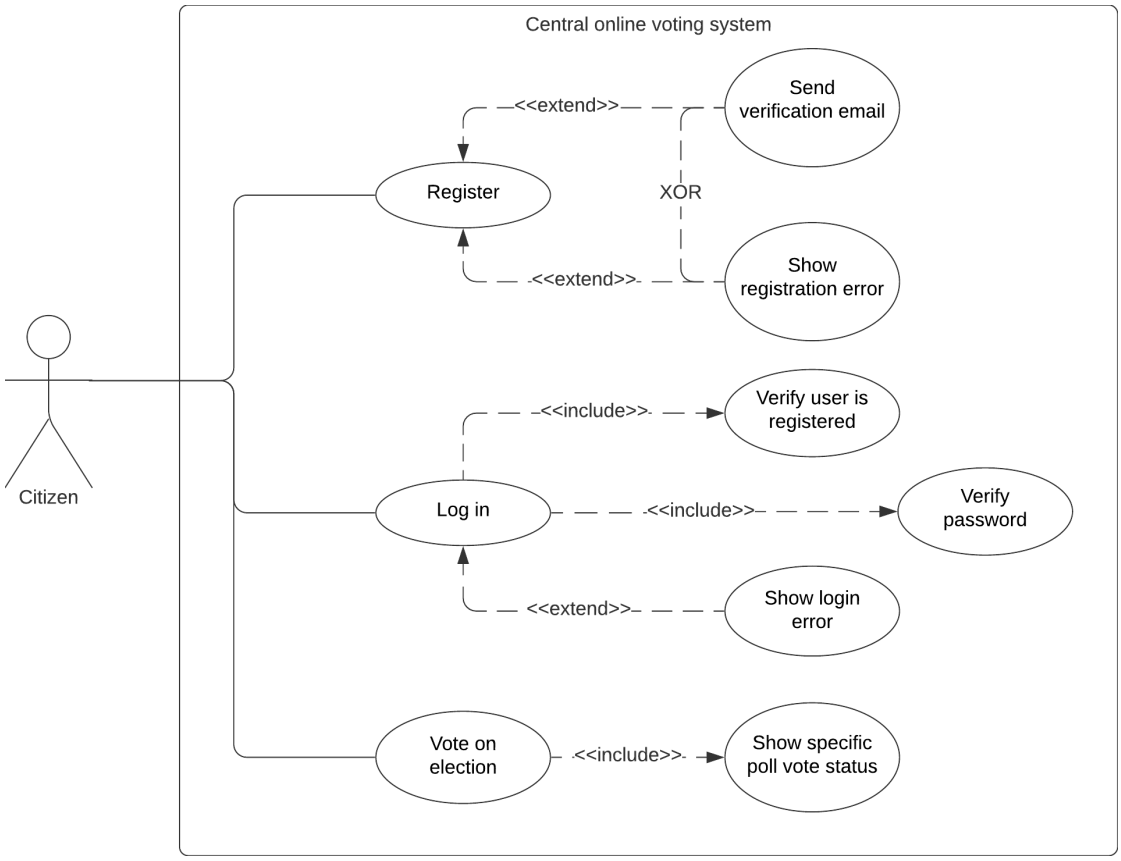

Figure 3.1: Citizen use case UML for Central online voting system

Figure 3.1 shows available use cases for citizen. He can register and in that case he will receive an email, or he can see some error regarding registration. Citizen can try to login, which will trigger checks if he is registered and that password matches. In case of any errors, he will see a message. Citizen can vote, and after that he will see status of his vote regarding every poll in the election, no matter if vote is valid or not. [Czy użytkownik, który się jeszcze nie zalogował «citizen» może głosować?]

3.3.2 Ward administrator use case UML

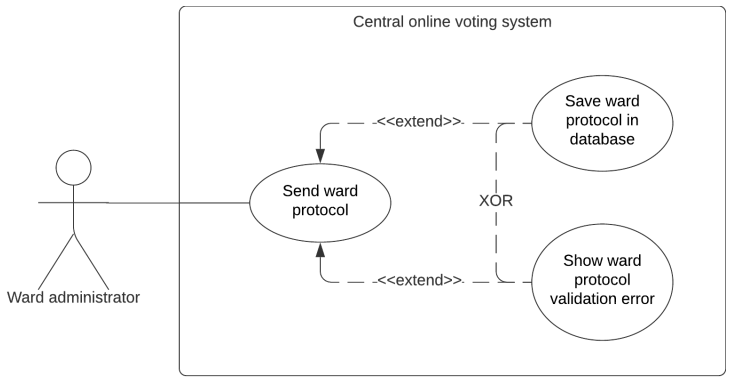

Figure 3.2: Ward administrator use case UML for Central online voting system

Figure 3.2 shows available use cases for a ward administrator. A ward administrator can try to send a ward protocol. If anything is not correct in a protocol he tried to send, he will see a validation error, and the protocol will not be sent.

3.3.3 Committee administrator use case UML

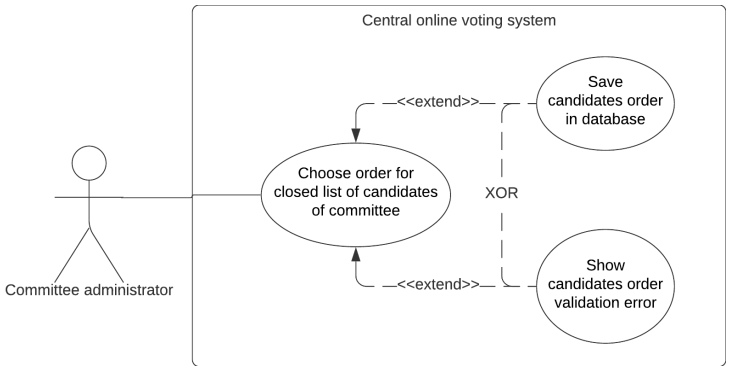

Figure 3.3: Committee administrator use case UML for Central online voting system

Figure 3.3 shows available use cases for a committee administrator. A committee administrator can send closed list candidates' order. If anything is not correct in an order of candidates, he will see a validation error, and the candidates order will not be saved.

3.3.4 Administrator use case UML

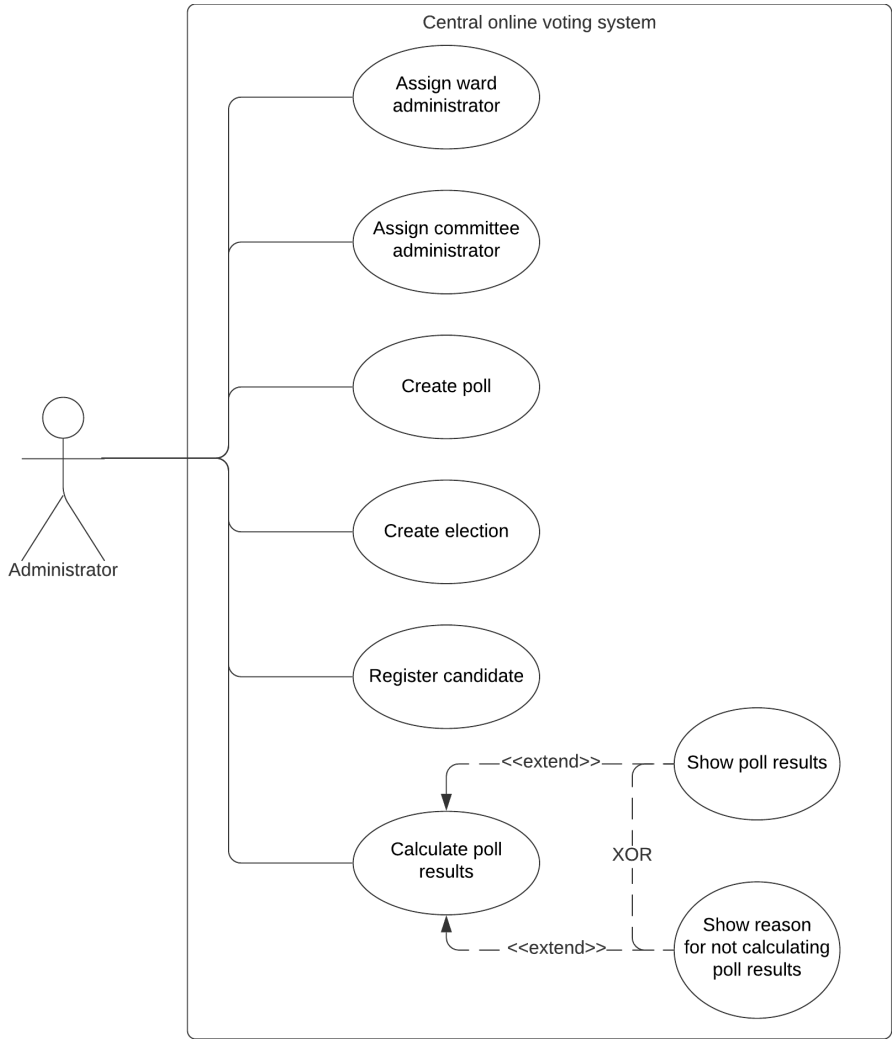

Figure 3.4: Administrator use case UML for Central online voting system

Figure 3.4 shows available use cases for an administrator. We assume that an administrator knows what he is doing, so there is no need to validate his choices. However, he has no power over how many protocols are sent from wards when he wants to calculate poll results. Thus he will get appropriate message when he tries to calculate results of poll that is not ready to be calculated.

3.4 Description of tools

The application is developed in Java programming language with Spring Framework and Thymeleaf as template engine. Data is stored in a MySQL database, and whole application is built with Apache Maven build automation tool.

3.4.1 Java

It's [It is] an object–oriented programming language that requires Java Virtual Machine to run a program. This allows Java applications to run on almost every device, regardless of it's operational system[4]. Java offers huge open–source ecosystem.

Alternative

C# is also an object–oriented programming language, but it's [it is] used to develop software for Microsoft platforms[spacja przed cytowaniem][11].

Although there is no much difference in beginner level development between Java and C#, I find C# less convenient than Java, because I use machine with macOS operating system as my everyday work tool, and Java has huge support on Unix based system, compared to almost none support for C#.

3.4.2 Maven

Apache Maven is a tool that is capable of building and managing any Java-based project. It helps developers with dependency management, as it's able to download all necessary dependencies with almost no participation of developer. It's easy for beginners, as it's build lifecycles and syntax is easy to understand—all You need is artifact info of dependency that You want to include in Your project, and it will download and include dependency in project during install build lifecycle[2].

Alternatives

Gradle It's also application build tool for Java-based projects. It's better than Maven for larger projects, thanks to better automation methods and better per-

formance. Gradle is built with more experienced and demanding developer in mind[5].

Ant Ant is a powerful tool that allows developer to automate very complicated task, but this functionality is not cheap. It's complicated to configure and it's being ousted by Gradle[1].

I chose Maven because I wanted to learn modern build technology that is common in the industry, but is suitable for beginners.

3.4.3 Spring

It's an open–source framework for Java platform. It focuses on delivering out–of–the–box core features for any kind of application, so developers can focus on business logic of the application. Aside from Spring Core, which provides easy dependency injection, Spring provides extensions in form of more or less independent modules, each available as it's own maven artifact.[7] Spring Boot is an extension of Spring framework, which provides out–of–the–box configuration, and requires user to provide only application specific configuration.

Alternative

Because Spring is open–source framework, considered a standard in the industry for over 10 years, there is no other framework that is as feature–packed as Spring. One could make a case for .Net framework for C# language, but it's developed and directed by Microsoft[11], whereas Spring is powered by community, and contains features required by that same community.

Spring Security

Authentication and Authorization framework for Spring based applications. Provides quick protection of API exposed by application and integration with users in database. Allows for assigning users' roles, and managing accesses based on these roles[8].

Spring Web MVC

It's designed around front controller pattern, and helps developers follow the Model–View–Controller design pattern. It coordinates flow of request processing performed by configurable delegate components[9].

Spring Data JPA

Spring implementation of JPA contract, powered by Hibernate. Reduces hugely boilerplate code that's associated with querying database, especially with Repository interface custom finder methods[6].

3.4.4 Thymeleaf

Modern server-side Java template engine resolver. It converts HTML files with Thymeleaf–specific dynamic attributes based on model attributes provided by developer through Spring MVC framework[10].

Alternative

There are not many pure alternatives when it comes to template resolvers. Java Enterprise Edition comes with JSP, which is not pleasant to use, and is considered obsolete. It definitely does not provide level of integration with Spring MVC framework as much as Thymeleaf does.

3.5 Methodology of design and implementation

Base of every application is a well designed database schema. Once schema is considered with required functionalities of application in mind, and with space for extensibility predicted, it makes development not only go faster and more efficient, but also more pleasant for a developer.

With proper database schema, we have to consider what is the best way for user to use our application—through web browser or application? Because application is meant to be nation–wide, and there is a web browser on perhaps every com-

puter with internet connection—which is necessary for application to work—web application is the way to go.

When designing web application, we have to decide how we want to achieve end result—presenting view to application user. There are two most common approaches when it comes to Spring application:

- Exposing REST API from Spring application, that contains all of business logic and create another application, that consumes that API and interacts with the user in any way desired.
- Keep view rendering on server side, and expose pure html to the user.

I chose the second option, because it is not the most important thing for Central online voting systemto be pretty and responsive, I wanted to focus on backend side of the application.

Once view presenting is established, we want to consider how we want to design architecture of the application. I can see two approaches in choosing the architecture: basing on application requirements or basing on technology used in application. Because Spring provides whole framework dedicated to MVC design pattern, and also it's widely used in the industry, I chose to develop application in that manner, rather than choosing architecture basing on application requirements. My application is not large enough to be hugely impacted by bad architecture. I could only harm my project by using some fancy architecture.

Chapter 4

External specification

-

Chapter 5

Internal specification

-

Chapter 6

Verification and validation

-

Chapter 7

Conclusions

-

Bibliography

[1] The Apache Software Foundation. The apache ant project.

[2] The Apache Software Foundation. Apache maven project.

[3] S. Heiberg and J. Willemson. Verifiable internet voting in estonia. In *2014 6th International Conference on Electronic Voting: Verifying the Vote (EVOTE)*, pages 1–8, 2014.

[4] C.S. Horstmann and G. Cornell. *Core Java, Volume I–Fundamentals*. Sun Core Series. HELION S.A., 2008.

[5] Gradle Inc. Gradle build tool.

[6] Inc. Pivotal Software. Spring data jpa.

[7] Inc. Pivotal Software. Spring framework overview.

[8] Inc. Pivotal Software. Spring security.

[9] Inc. Pivotal Software. Web mvc framework.

[10] The Thymeleaf Team. Thymeleaf.

[11] T. Thai and H. Lam. *.NET Framework Essentials*. Essentials Series. O'Reilly Media, 2003.

Appendices

List of abbreviations and symbols

API	Application Programming Interface
HTML	HyperText Markup Language
MVC	Model–View–Controller
REST	Representational State Transfer
UML	Unified Modeling Language
JPA	Java Persistence API
JSP	JavaServer Pages

Listings

Contents of attached CD

The thesis is accompanied by a CD containing:

- thesis (LaTeX source files and final `pdf` file),
- source code of the application,
- test data.

List of Figures

List of Tables