MINISTRY OF EDUCATION AND SCIENTIFIC RESEARCH

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

CONFERENCE MANAGEMENT SYSTEM

CONFERENCE MANAGEMENT SYSTEM

Graduate: **Alina Ioana MOLDOVAN**

Supervisor: **as. ing. Cosmina IVAN**

2017

MINISTRY OF EDUCATION AND SCIENTIFIC RESEARCH

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

DEAN,
Prof. dr. eng. Liviu MICLEA

HEAD OF DEPARTMENT,
Prof. dr. eng. Rodica POTOLEA

Graduate: **Alina Ioana MOLDOVAN**

CONFERENCE MANAGEMENT SYSTEM

1. **Project proposal:** *The project aims at implementing a system that deals with conference management. The system aims to provide the reduction of the time necessary to execute tasks like assignment of papers to reviewers, conflict of interest detection, plagiarism detection*

2. **Project contents:** *Table of contents, Introduction, Project Objectives, Bibliographic Study, Analysis and Theoretical Foundation, Detailed Design and Implementation, Testing and Validation, User manual, Conclusions and Further Developments, Bibliography and Annexes.*

3. **Place of documentation**: *Technical University of Cluj-Napoca, Computer Science Department*

4. **Consultants**: as. ing. Cosmina IVAN

5. **Date of issue of the proposal:** November 1, 2016

6. **Date of delivery:** July 14, 2017

Graduate: _____

Supervisor: _____

MINISTRY OF EDUCATION AND SCIENTIFIC RESEARCH

TECHNICAL UNIVERSITY
OF CLUJ-NAPOCA

FACULTY OF AUTOMATION AND COMPUTER SCIENCE
COMPUTER SCIENCE DEPARTMENT

Declaraţie pe proprie răspundere privind autenticitatea lucrării de licenţă

Subsemnatul(a) Alina Ioana Moldovan, legitimat(ă) cu cartea de identitate seria AX nr. 482082, CNP 2940712011851, autorul lucrării Conference Management System, elaborată în vederea susţinerii examenului de finalizare a studiilor de licenţă la Facultatea de Automatică şi Calculatoare, Specializarea Calculatoare in limba engleza din cadrul Universităţii Tehnice din Cluj-Napoca, sesiunea Iulie a anului universitar 2016-2017, declar pe proprie răspundere, că această lucrare este rezultatul propriei activităţi intelectuale, pe baza cercetărilor mele şi pe baza informaţiilor obţinute din surse care au fost citate, în textul lucrării, şi în bibliografie.

Declar, că această lucrare nu conţine porţiuni plagiate, iar sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor.

Declar, de asemenea, că această lucrare nu a mai fost prezentată în faţa unei alte comisii de examen de licenţă.

In cazul constatării ulterioare a unor declaraţii false, voi suporta sancţiunile administrative, respectiv, anularea examenului de licenţă.

Data

Nume, Prenume

_____

_____

Semnătura

Table of Contents

Chapter 1. Introduction

At this moment, in any field, the efficiency of the activities, reducing the time necessary to meet various working resources is pursued. This trend lead to the translation of many activities and the attempt to solve them with the support of various general or specialized computer systems. There has been a rapid spread of technology, that required the development of increasingly efficient and competitive computer systems. Reducing processing time, costs, and necessary human resources is what is meant to be accomplished through automation and a transfer of activities into virtual space.

1.1. Project context

Conference organization and management is a process that is anticipated with great apprehension by many people that go through it. One of the main reason is the amount of work, detail that needs to be taken into consideration, the importance of keeping up to deadlines, the exchange, storage and processing of huge amount of data, management of special requests, registration and many other details [1]. Academic conferences play a key role in exchanging researchers current workings. Many academic conferences are held annually which lead to an increase in the number of submitted papers and the amount of work necessary to manage the submissions and review process. Such an intricate work flow of conference management results in this process being anticipated with great apprehension [2].

In their early days, web based conference management system only offered basic functionality like the use of web page to display information about the conference and announcements. In the last several years' conference management systems have become an important aspect when organizing a conference.

As web based information systems, they offer a user-friendly service, help authors track their papers, allow program committee members review the papers anywhere in the world. The most important benefit offered by these type of systems is not only the user-friendly way of communication and data storage, but the automation of a set of hard to handle and time-consuming processes like assigning papers to reviewers, resolve the conflict of interest and plagiarism detection. [3].

In order to satisfy the requirements and demands, an online conference management system should offer all the basic functionality necessary to manage a conference and besides that it should provide new approaches for functions and introduce new functionalities.

1.2. Motivation

Taking into consideration the time and human resources necessary, in order to reduce them the idea to translate these tasks into a virtual and compact system appeared. Studying the systems for conference management that are available on the market at the current moment it has been noticed that these systems cover the process for management, offers support for multiple users and some for a conference schedule.

The requirement for these type of system that integrate a large number of tasks from the conference management process and other activities, has increased. The systems that are available on the market at the current moment, provide a level of automation that can be further extended and new elements can be integrated. These represented the premises that lead to choosing these theme for the bachelor thesis that has as main purpose to cover the existing functionality, integrate new elements and provide a high degree of satisfaction for the user.

The usage of these type of systems offers advantages to all categories of users. The program chair can supervise the organization process from any corner of the word, the author can review the submission from anywhere and reviewers can be invited from any corner of the world.

As a result of the bibliographical study the expected result of this project is an application that covers the basic functionality of a conference management system, offers new approaches for the necessary activities and introduces new components.

1.3. Paper content

In the next paragraphs the structure of the thesis, the chapters and their content is presented

Chapter 1 Introduction - This chapter presents a short description of the context of the problem that it wants to be resolved through the system.

Chapter 2 Objectives of the project – Contains the presentation and a short description of the main and secondary objectives that correspond to the application.

Chapter 3 Bibliographic study – This chapter will contain a short description of the concept conference management system and the principles that form this system. It will also contain a short comparison between the system that deal with conference management and the application implemented.

Chapter 4 Analysis and theoretical foundation – This chapter will contain a short description of the technologies that were used to implement this Web application. Besides that, in this chapter the functional and non-functional requirements that were identified are presented, also this chapter will also contain the significant use case that were identified are presented.

Chapter 5 Detailed design and implementation – In this chapter the architecture of the system will be presented and described in detail. This chapter will include class

diagram, deployment and package diagram. Each important component will be described in detail. At the end of this chapter the database schema will be presented, used in order ensure persistence of the data and to fulfill the functional and non-functional requirements of the system.

Chapter 6 Testing and validation – In this chapter the methods used for testing the application and the results obtained are presented. The testing and validation were done in order to test and validate some of the functional and non-functional requirements of the system

Chapter 7 User's manual - This chapter will present the software requirements in order to install the system. Also in this chapter, the necessary instruction in order to install and use the system are presented.

Chapter 8 Conclusions and further development – In this chapter the objectives that were fulfilled thorough the implementation of the system and possible improvements that can be applied are presented.

Chapter 2. Project Objectives

This chapter will outline the objectives set to be achieved. The system is designed not only to cover the conference management process, but also present new elements and introduce new components

2.1. Primary objectives

The primary objective of this project is to define, implement and build a system which offers support for the conference management, that can be fulfilled only thorough achieving the secondary objectives that will be presented next.

2.2. Secondary objectives

One of the objectives of this project are to enhance the efficiency in usage, that is measured through the expressivity and the consistency of the graphical user interface. A user is considered to be efficient when using a system if the time necessary to execute a certain task decreases with each usage.

Another objective is to create a system that allows further improvements, extensions of the current functionality. The system should be able to offer the opportunity to manage conferences, submissions, reviewers, available sessions, available tracks or the schedule of the conference.

The system will be used by more than one category of users. The available categories of users will be program chair that also plays the role of a system administrator, author that will submit a paper to the system and wait for a decision and reviewers. Each of these categories of users will have a set of functionalities that the system will provide, which will be considered as objectives. Different types of users imply different levels of access to the system resources, one of the objectives being the capacity to access the resources available based on the role of the user that makes the request.

Paper Management

The system should offer the possibility to every user to submit a paper to the conference, to edit that submission. From the moment when the submission is recorded to the system, the user is capable of tracking the status of the paper and request information's about it.

Conference Management

The system should offer to the user the capability to organize, create a conference, offer information's about it. From the moment the conference is submitted to the system, the organizer should open submissions, invite reviewers, close submissions, assign papers to reviewers and inform the authors about the status of their papers

Topic and multi-track management

The system should offer to users after the conference was created the capacity to select and assign a certain topic. Have the possibility to create and assign to conferences more than one track. Once the track is created and registered to the system the user will have the power to update it.

Payment submission

The system should offer to users the capability to buy a ticket to the conference. Once the sale is registered into the system the user will have a limited time to edit or cancel it. The desired method of payment will be implemented with credit cards.

Conflict of interest management and plagiarism management

The system should provide the capability to detect a conflict of interest in order to avoid the apparition of favoritism. Should also provide the capability to detect work that is not original in order to avoid plagiarism. For the plagiarism part once a submission is recorded the system will automatically pass it through a service that will compare it to past publications that are indexed in that service. For the conflict of interest detection once a paper is assigned to a reviewer and stored, the system will automatically check the information provided by the user to the information provided in the submission and signal a conflict if there are some information, like same institution.

Paper/Review assignment

The system should provide the capability to assign a paper to a set of reviewers. Once the submission period has ended the assignment part should begin. The assignment part consists in using an automated manner, association realized based on some keywords introduced by the reviewer and the keywords introduced in the submission, or based on the previous published papers.

Review Management

The system should provide to reviewers the capability to accept/refuse a paper, to create a review based on the paper assigned to it. From the moment, the review is stored in the system the user will have the capability to edit it or remove it.

Announcements management

This system will provide an automatic announcements service. The announcements will be delivered under the form of emails. The delivery of emails will be made using an external service. The system should provide the capability to send announcements. But once the data will be stored in the system the user will have the capability when editing to resend it, or simply remove the announcement from the system

Chapter 3. Bibliographic Research

In this chapter a set of similar systems that are popular in conference management are presented. An important part of the bibliographic study consisted in analyzing the available functionality of these systems and identify the functionality necessary for organizing a conference.

3.1. Developing web applications

When developing a web application, the main concerns should be device, software independence but also scalability. A web application is nothing more than presentation environment and access to resources, where besides the audio, photo, video resources the web pages are also considered to be resources. The access to these resources is made through an address Uniform Resource Identifier (URI) and a communication protocol Hypertext Transfer Protocol (HTTP) which sustains the access requests made through the URI. A web application represents an interconnected collection of web pages with dynamic content, developed to offer a specific functionality. The interaction user application is made though the web interface.

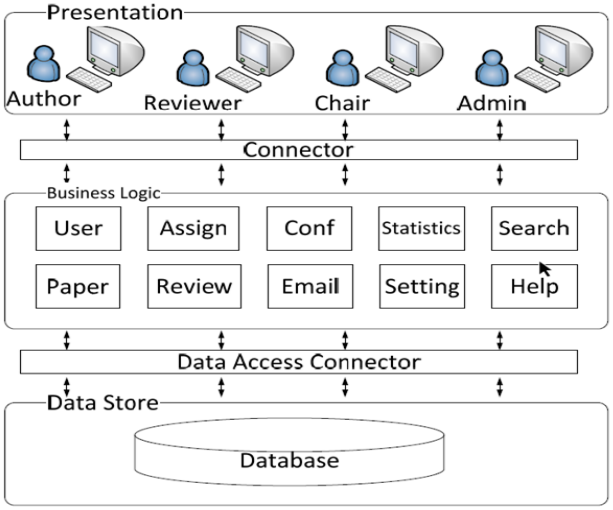

Figure 3-1 General conference mangment system architecture on the Client Server model [4]

The architecture of any web application is based on the Client-Server architecture, and the mapping of these two components is very simple. If the web browser is considered to be the Client, through which the request for resources is made. The Server considered to be the one that responds to the requests that the client makes. The web browser will parse the data that is introduce by the user and determine the communication protocol, usually HTTP, the name of the server and the resource that the user wants to access. After this, the browser will open a connection to the server sending a message which contains the operation that it should be executed. After sending this message, the server verifies if the resource desired is available and can be accessed. In any context, the client receives a

message from the server that contains the status and the content of the accessed resource. An example of this architecture can be seen in Figure 3-1

3.2. Conference management

Conference management relates to the executive management of a conference either in-house within a company or for a client of a professional conference organizer (PCO). It constitutes of the basic management tools that involve planning, organizing, leading and control. Arranging a conference from conceptualization to execution can take any time between 17 days (professionals in the field of conference production) to almost 12 or 18 months.

There exists a variety of different conference that ranges from small scale executive meetings to international summits that caters for up to 65 000 people at a time depending on the size of the venue (e.g. stadiums) The most common form of conferencing is based on industry and trade that uses the opportunity to network and benchmark industry trends and standards. The conference management industry is part of the tourism industry and can be used to generate revenue for a community, town, country or region. Management, organization and creation are processes that are similar for conferences in many respects. For academic conferences, there is an additional requirement which is the existence of a review process.

For this system, the conference management process for an academic conference was taken into consideration. In order to determine the requirements for conference management, it is necessary to analyze the stages of this process in detail. A generalized process is described below which present the main phases that a conference management and organization requires.

The conference organization starts with the formation of steering committee, that take the major decision of the conference. The conference process starts in earnest with the formation of a steering committee. This is the main committee that takes major policy decisions about the conference. The steering committee meets well in advance of the conference and develops its plans far ahead of the event. One of the main duties of this committee is to form other committees to run the organization activities. These include the organizing and the technical (scientific) committees. These major committees themselves may form the necessary sub-committees in order to carry out the groups of tasks. The technical (scientific) committee itself may form the following sub-committees: technical program and reviewer sub-committees. The conference technical areas and topics are usually determined at the joint organizing and technical committee meetings and are announced in the call for papers (CFP) announcements. Other organizational issues and deadlines are determined by the organizing committee. Although there may be wide variations in the phases of conferences, a widely found phasing of academic conferences is as follows:

I. planning and organization; leading to announcement and CFP,
II. manuscript submission,
III. review process; reviewer assignment and
IV. paper review,

V. announcement of review results,
VI. camera ready paper (CRP) submission,
VII. registration,
VIII. pre-conference operations,
IX. the conference-event; execution of the
X. conference, and
XI. post-conference activities and evaluation

During the conference life-cycle various events can occur that trigger the ending and/or start of a new phase. The phases and important events in the academic conference are shown in Figure 3-2. The existence and the duration of a certain phase may vary according to the particular conference.

However, each of the events will be presented in the conference organization and management in some form. During each phase, activities for the up-coming phases and events will be planned and carried out. Various processes are carried out within the conference life-cycle and within each of the phases described above. Figure 3-2 presents a typical conference process with the committees set up and the associated processes and activities carried out. Furthermore, certain activities may be delegated to sub-committees, hence changing the structure of the chart. The process diagram is self-explanatory and summarizes all of the main activities that need to be planned for a successful academic conference. An important part of any current conference planning and organization is the information technology (IT) related plans and logistics needed to be employed [1].

Figure 3-3 presents an overall description of the conference management process, which helped identify the necessary phases and provide a better organization.

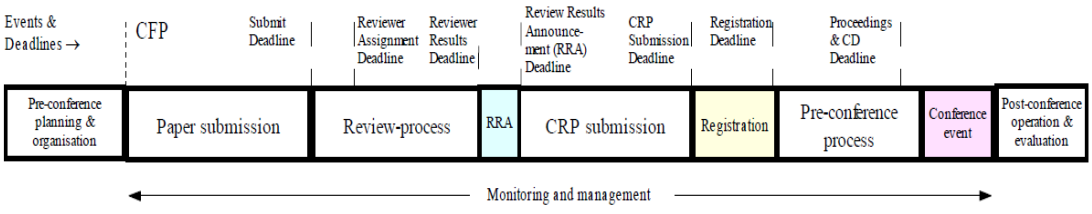

Figure 3-2 Academic conference life-cycle [1]

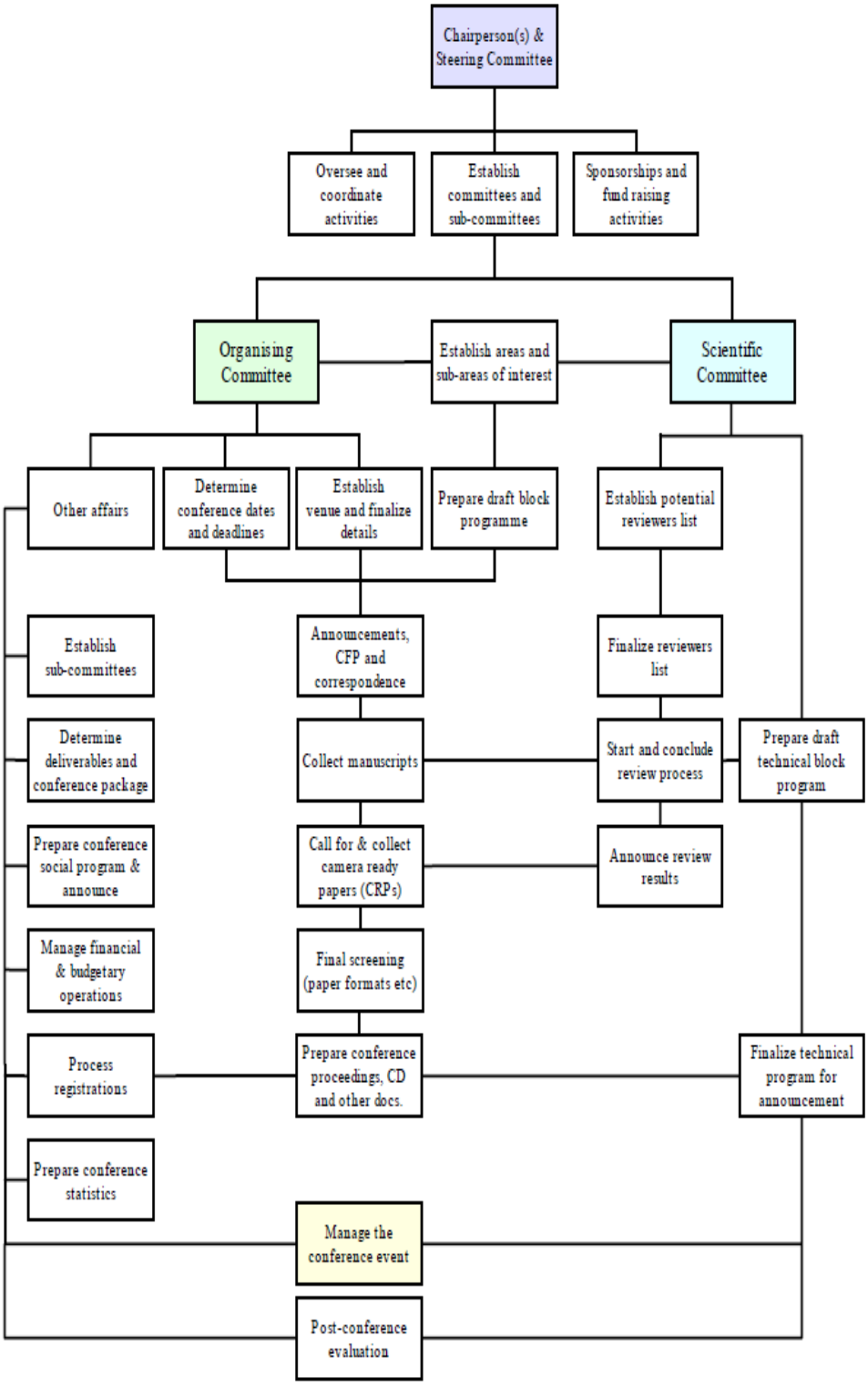

Figure 3-3 Typical academic conference organization and process chart [1]

3.3. Similar systems

In accordance with the current trends in computer software, most of the system analyzed are offered not as a traditional sellable software product but as a completely hosted service. The *Software as a Service (SaaS)* solutions minimize the total cost of service by releasing conference organizers from buying any hardware and software, hiring system administrators and programmers to install and support them and etc. Beside the differences in graphics and user interface all conference management systems provide all the basic functionalities needed to manage the processes of paper submission, assignment of reviewers to papers, review submission and etc.

Most systems offer an automatic pre-press processing of papers and conference proceedings, automatic generation of conference program (based on manual paper clustering), authors' index and table of contents [5]. After the bibliographic study, the comparison was restricted to 4 systems, the first one also being the popular one at the current moment EasyChair and ConfTool, Microsoft conference management toolkit and MyReview

3.3.1. EasyChair

EasyChair [6] is a free web-based conference management software system used, among other tasks, to organize paper submission and review. Users include top conferences in several areas, for example, in World Wide Web and in Bioinformatics. Users range from small workshops with around 10 submissions to big conferences having thousands of submissions.

The first version of EasyChair was implemented in 2002. It was used by 12 conferences in 2002-2004 and by 66 conferences in 2005. It is considered to be since 2006 number one conference management system in the number of conferences, users and submissions. All together EasyChair proudly hosted **53,739** conferences and served **1,954,080** users. EasyChair offers three types of licenses: executive, professional and free. These types of licenses differ through the features offered by it.

Table 3-1 EasyChair functions

Feature	Description
FR 1	conflict of interest
FR 3	multi-track support
FR 4	manual submission assignment to reviewers
FR 5	automatic submission assignment to reviewers
FR 6	advanced conflict management
FR 7	email to authors and program committee

3.3.2. ConfTool

ConfTool [7] is a Web-based event management system developed to support the organization of academic conferences, workshops, congresses and seminars. It has helped more than 1000 organizers to make their events a success and is available in over 10 languages. It offers two versions for different requirements. The standard version VSIS ConfTool was designed for smaller events with up to 150 participants. It is an open/shared-source system available under different licenses. VSIS ConfTool is geared towards academic organizers of small, non-commercial events. The system features only basic functions and is offered without support and for local installation only. It is offered a free license on request. The professional version ConfTool Pro has substantially more features, is more flexible and also suitable for events with many participants, several contribution types and/or sub-events. ConfTool Pro is offered as a hosted service with full technical support. The fee depends on the requirements and the size of the event.

Table 3-2 ConfTool functions

Feature	**Description**
FR 1	Multi-Track Support
FR 2	Automatic Identification of Conflict of Interest
FR 5	Reviewers Can Refuse Assignments
FR 7	Invitations for Reviewers
FR 8	Manual Assignment of Reviewers
FR 9	Automatic Assignment of Reviewers

3.3.3. Microsoft's Academic Conference Management Service

Microsoft's Academic Conference Management Service [8] is a free and hosted conference management service sponsored by Microsoft Research. CMT is built using Microsoft technologies including Azure Cloud Platform, ASP.Net, Internet Information Server (IIS) and SQL Server. All conferences' data are geo-replicated to provide additional redundancy and great service availability. CMT's functionalities are fully accessible through web based interfaces. CMT provides rich support to the Program Committee chairs for managing the conference workflow including customization of conference properties e.g., multiple tracks, deadlines, author submission and reviewer forms, double-blind reviewing, allowing authors to mark conflicts of interest with reviewers, use of external reviewers and meta reviewers. CMT also provides filtering, sorting and aggregation functionality as well as emailing capability to authors and reviewers that makes it easy to handle conferences with a large number of reviewers and

submissions. CMT's extensive auditing capabilities allow Program Committee Chairs to track accesses to conference data. CMT enables the Program Committee chairs to easily bulk-import data as well as bulk-export conference data and reports in popular formats such as XML and Excel. It is capable of handling the complex workflow of an academic conference

Table 3-3 Microsoft's Academic Conference Management Service functions

Feature	**Description**
FR 1	Multi-track support
FR 2	Automatic and manual assignment of papers reviewers
FR 3	Advanced conflicts management
FR 6	Integration with iThenticate plagiarism detection service
FR 7	Creation of sessions for accepted papers
FR 8	Integration with Toronto Paper Matching System to perform paper reviewer matching

3.3.4. MyReview

MyReview [9] is a web-based conference management software which has been implemented in May-July 2003, and initially used for managing the ACM conference on Geographic Information Systems (ACM-GIS), is a paper review system written with PHP and MySQL. It proposes the traditional functionalities of such systems namely paper submission, reviewer assignment, discussion on conflicting reviews, selection of papers, mails sent to all actors, etc. One of its most salient features, though, is the ability to change the look-and-feel of the HTML pages, as well as all the static, textual information, independently from the PHP code. To do so, it relies on the templates mechanism of PHP which separates (almost) completely the PHP code from the HTML/images/JavaScript/Flash presentation.

The main goals of the system are to provide and easy-to-install and easy-to-manage software, based on the most up-to-date and widespread technologies. It proposes the traditional functionalities of such systems

Table 3-4 MyReview functions

Feature	**Description**
FR 1	Paper submission
FR 2	Manual assignment of papers to reviewers.
FR 3	Automatic assignment of papers to reviewers
FR 4	Review submission.
FR 6	Paper selection.

3.3.5. Comparison table

Table 3-5 System comparison [10]

No	**System Feature**	**EasyChair**	**CMT**	**My Review**	**System**
1	Register to system	Yes	Yes	Yes	Yes
2	Create conference	Yes	Yes	Yes	Yes
3	Dates and deadlines.	Yes	Yes	N. A	Yes
4	Specify topics for conference	Yes	Yes	Yes	Yes
5	Define Roles-chair, authors, reviewers	Yes	Yes	Yes	Yes
6	Profile: Change your account, show your profile, change your password, upload a photo	Yes	N. S	N. S	Yes
7	Submit Paper to a valid conference.	Yes	Yes	Yes	Yes
8	Lists the conferences where the user is a chair	Yes	Yes	Yes	Yes
9	View all the papers submitted by the user. Including accepted, rejected, published and withdrawn	Yes	Yes	Yes	Yes
10	View details of submitted papers	Yes	Yes	Yes	Yes

11	Conflict of interest	Yes	Yes	Yes	Yes
12	Paper assignment	M/A	M/A	A	A
13	Review forms	S	D	D	S
14	Schedule of the conference	Yes	N. S	No	Yes
15	Multi-track service	Yes	Yes	No	Yes
16	Send conference related mails to registered users	Yes	Yes	Yes	Yes
17	Submissions format	Multiple formats	Multiple formats	Abstract	Multiple formats
18	Transaction based operations	Yes	Yes	Yes	Yes
19	Backup database	N. A	Yes	N. S	Yes
20	Plagiarism check	No	Yes	No	Yes
21	Review papers	Yes	Yes	Yes	Yes
22	Charges	No	Yes	No	Yes

The analysis was made on a set of criteria's that were identified as common to the evaluated systems, but there is a set of functionalities that are specific to the developed system and they are presented next

- Automation of the assignment of reviewers to papers, which will reduce the amount of time and resources necessary, which will speed up the management process
- Resolution of the conflict of interest
- Addition of a plagiarism detection function that will reduce the time necessary for review by identifying the papers that are not original.
- Addition of a payment processing system that will deal with the necessary payment taking some of the burden from the organization committee
- Reset password functionality

Chapter 4. Analysis and Theoretical Foundation

In this chapter, the technologies that were used to implement this web application will be presented. They are: Java Standard Edition, Spring framework, MongoDB etc. It describes the necessary details for integrating with the source code, with the necessary references to the technical literature that details the mentioned technologies.

4.1. System requirements

System requirements can be classified as functional and non-functional requirements. Functional requirements present a complete description of how the system will function from the user's perspective. Non-functional requirements, in contrast, dictate properties and impose constraints on the project or system. They specify attributes of the system, rather than what the system will do [11]

The functional and non-functional requirements are put together according to Grady Booch's FURPS+ classification:

- **F**unctional capabilities
- **U**sability
- **R**eliability
- **P**erformance
- **S**upportability
- **S**ecurity

4.1.1. Functional requierments

Functional requirements present a complete description of how the system will function from the user's perspective. For a better management of the system requirements, these will be presented based on the category of users that can access them. [11]. The set of functional requirements identified for the system are presented in Table 4-1

Table 4-1 System functional requierments

Number	Functional requirement	User
FR 1	User account- forgot password	All Users
FR 1.1	Update user information	All Users
FR 1.2	Upload picture	All Users
FR 2	User management	Program chair
FR 3	Conference management	Program chair
FR 3.1	Add topic to the conference	Program chair

FR 3.2	Add tracks to the conference	Program chair
FR 3.3	Add sessions to the conference	Program chair
FR 4	Handle notifications	Program chair
FR 5	Handle multi-tracks	Program chair
FR 6	Invite reviewers	Program chair
FR 7	Assign papers to reviewers	Program chair
FR 8	Create conference program	Program chair
FR 9	Create booklet	Program chair
FR 10	Create e-proceedings	Program chair
FR 11	Plagiarism detection	Program chair
FR 12	Info authors	Program chair
FR 13	Handle submission	Author
FR 13.1	Display submissions	Author
FR 13.2	Update submission	Author
FR 13.3	Review paper status	Author
FR 14	Accept/ Reject papers	Reviewer
FR 15	Review paper	Reviewer
FR 16	Register	Visitor
FR 17	View accepted papers	Visitor
FR 18	Buy ticket	Visitor

4.1.2. Non-functional requirements

Non-functional requirements, dictate properties and impose constraints on the project or system. They specify *attributes* of the system, rather than what the system will do.

Simplicity: the activities should be simple and consistent. The activities will be directed through simple links and simple dialogues. The system will also provide some help through error messages, remarks, and documentations.

User-friendly: the interface of the system should be user-friendly. It should be easy to use or learn. The interface of the system should also not require expert users to understand.

Accessibility: Conference Management system will be available for all users through internet. No special features are considered for people with disabilities or special needs.

Security: the system will allow permissible information flow after authenticating the identity of the actor on the system. In addition, the user's information should be confidential. The system will identify each user by his ID, which will be his username, required to be unique, with a password that is stored in an encrypted way. Encryption method described in the section *4.2.3.3 Salted password hashing*. Authors are not allowed to submit their papers unless they are registered users in the CMS. The information related to user's activities will be stored in the database

Availability: The system will be functioning properly all time for all users except for cases of failure that could be recovered within 1 hour.

Reliability: if the system maintains its performance over time, then it will be considered reliable. The application should recover within not more than 1 hour in case of a system failure during the rush hours.

Performance: There are no special constraints on the performance. This requirement measures the efficiency, speed, and throughput of the system. The Conference management system will make any transaction on the system within less than 10 seconds depending on the internet/machine speed such as:

- Login to the system
- Open any of the CMAS pages
- Sending message to chair, or from chair to registered users.
- Submit new paper.
- Submit review form.
- Make the final decision on a paper.
- Register for conference.
- Getting some reports by the chair such as: list of all accepted papers, list of all registered authors, and so on.
- Execute other operations

Maintainability: creating new system that can be adjusted easily for adopting new technologies or to fix any defect is called maintainability. The CMS has been analyzed in a way that can handle new or future improvements easily. The database can be exported (as a backup copy) or imported to the system. The system helps the users in reducing usage errors by providing some hints to the users. Moreover, validating the input data before submission make the system more immune against faults [2]

4.2. Technologies used in developing the web application

4.2.1. Java Standard Edition

Java is a is free, simple, object-oriented programming, distributed, supports multithreading and offers multimedia and network support. Java Standard Edition (Java SE) allows develop and deploy Java applications on desktops and servers, as well as in today's demanding embedded environments. Java offers the rich user interface, performance, versatility, portability, and security that today's applications requires [12]. Chosen due to the performance, versatility, ease of use, security and multithreading support.

4.2.2. Maven

Maven is a build automation tool used primarily for Java projects. Maven addresses two aspects of building software: first, it describes how software is built, and second, it describes its dependencies. An XML file describes the software project being built, its dependencies on other external modules and components, the build order, directories, and required plug-ins.

This technology was chosen due the fact that Maven dynamically downloads Java libraries and Maven plug-ins from one or more repositories such as the Maven 2 Central Repository, and stores them in a local cache. This local cache of downloaded artifacts can also be updated with artifacts created by local projects. [13]

4.2.3. Spring framework

The Spring [14] framework is a Java platform that provides comprehensive infrastructure support for developing Java applications. Spring handles the infrastructure so the focus of the developer can be on the application. Spring enables the build of applications from "plain old Java objects" (POJOs) and to apply enterprise services non-invasively to POJOs. This capability applies to the Java SE programming model and to full and partial Java EE.

The Spring Framework *Inversion of Control* (IoC) component addresses development by providing a formalized means of composing disparate components into a fully working application ready for use. The Spring Framework codifies formalized design patterns as first-class objects that can be integrated in the application. Numerous organizations and institutions use the Spring Framework in this manner to engineer robust, maintainable applications

I have used Spring framework because it allows the presence of POJO classes, that are easily mapped to any database, easy integration with Maven and components that framework that allow a versatile application development.

4.2.3.1.*Spring boot*

Spring Boot [15] makes it easy to create stand-alone, production-grade Spring based Applications that can “just run”. Most Spring Boot applications need very little

configuration. Spring Boot can be used to create Java applications that can be started using java -jar or more traditional war deployments. Spring Boot provides a number of "Starters" that make easy to add jars to the class path. The *spring-boot-starter-parent* is a special starter that provides useful Maven defaults. It also provides a *dependency-management* section that contains a series of dependencies for which the version can be omitted.

Spring boot was chosen because it provides a radically faster and widely accessible getting started experience for all Spring development. Provides a range of non-functional features that are common to large classes of projects (e.g. embedded servers, security, metrics, health checks, externalized configuration). Absolutely no code generation and no requirement for XML configuration.

4.2.3.2. *Spring security*

Spring Security [16] provides comprehensive security services for Java EE-based enterprise software applications. There is a particular emphasis on supporting projects built using The Spring Framework, which is the leading Java EE solution for enterprise software development. Some familiarity with Spring - and in particular dependency injection principles - will help to get up to speed with Spring Security more easily. The two major areas of application security are "authentication" and "authorization" (or "access-control"), areas that Spring Security targets. "Authentication" is the process of establishing a principal is who they claim to be (a "principal" generally means a user, device or some other system which can perform an action in your application). "Authorization" refers to the process of deciding whether a principal is allowed to perform an action within the application.

4.2.3.3. *Salted password hashing*

Hash algorithms are one way functions. They turn any amount of data into a fixed-length "fingerprint" that cannot be reversed. They also have the property that if the input changes by even a tiny bit, the resulting hash is completely different. This is great for protecting passwords, because the passwords are stored in a form that protects them even if the password file itself is compromised, but at the same time, it is necessary to be able to verify that a user's password is correct.

The general workflow for account registration and authentication in a hash-based account system is as follows:

1. The user creates an account.
2. Their password is hashed and stored in the database. At no point is the plain-text (unencrypted) password ever written to the hard drive.
3. When the user attempts to login, the hash of the password they entered is checked against the hash of their real password (retrieved from the database).
4. If the hashes match, the user is granted access. If not, the user is told they entered invalid login credentials.
5. Steps 3 and 4 repeat every time someone tries to login to their account.

In step 4, never tell the user if it was the username or password they got wrong. Always display a generic message like *"Invalid username or password."*. This prevents attackers from enumerating valid usernames without knowing their passwords.

In order to ensure the security of the password, a hashed component, *salt*, which is generated using *Universally Unique Identifier (UUID)* generator available in Java. The *java.util.UUID* class represents an immutable universally unique identifier (UUID). Following are the important points about UUID: A UUID represents a 128-bit value. It is used for creating random file names, session id in web application, transaction id etc. There are four different basic types of UUIDs: time-based, DCE security, name-based, and randomly generated UUIDs.

The simplest way to crack a hash is to try to guess the password, hashing each guess, and checking if the guess's hash equals the hash being cracked. If the hashes are equal, the guess is the password. The two most common ways of guessing passwords are dictionary attacks and brute-force attacks. [17]

4.2.3.4. *Spring data MongoDB*

Spring Data MongoDB [18] project provides integration with the MongoDB document database. Key functional areas of Spring Data MongoDB are a POJO centric model for interacting with a Mongo Collection and writing a Repository style data access layer.

Chosen because of Spring configuration support using Java based @Configuration classes. MongoTemplate helper class increase productivity performing common Mongo operations. Includes integrated object mapping between documents and POJOs, exception translation into Spring's portable Data Access Exception hierarchy, feature Rich Object Mapping integrated with Spring's Conversion Service and annotation based mapping metadata.

4.2.3.5. *Spring Thymeleaf*

Thymeleaf [19] offers a set of Spring integrations that allows the usage as a fully-featured substitute for JSP in Spring MVC applications. These integrations will allow to make the mapped methods in the Spring MVC @Controller objects forward to templates managed by Thymeleaf, exactly like with JSPs. Create forms in templates that are completely integrated with form-backing beans and result bindings, including the use of property editors, conversion services and validation error handling.

This technology was chosen due to the easy integration with Spring, representing a substitute for JSP, use of Spring Expression Language.

4.2.4. NoSQL databases

NoSQL [21] is an approach to databases that represents a shift away from traditional relational database management systems (RDBMS). To define NoSQL, it is helpful to start by describing SQL, which is a query language used by RDBMS. Relational

databases rely on tables, columns, rows, or schemas to organize and retrieve data. In contrast, NoSQL databases do not rely on these structures and use more flexible data models. NoSQL can mean “not SQL” or “not only SQL.”

As RDBMS have increasingly failed to meet the performance, scalability, and flexibility needs that next-generation, data-intensive applications require, NoSQL databases have been adopted by mainstream enterprises. NoSQL is particularly useful for storing unstructured data, which is growing far more rapidly than structured data and does not fit the relational schemas of RDBMS. Common types of unstructured data include: user and session data; chat, messaging, and log data; time series data such as IoT and device data; and large objects such as video and images

4.2.4.1. *MongoDB*

MongoDB [22] is the database for today's applications enabling **fast, iterative development, flexible data model.** MongoDB's document data model makes it easy to store and combine data of any structure, without giving up sophisticated validation rules, data access and rich indexing functionality. The schema can be dynamically updated without downtime. Figure 4-1 presents the architecture of MongoDB system.

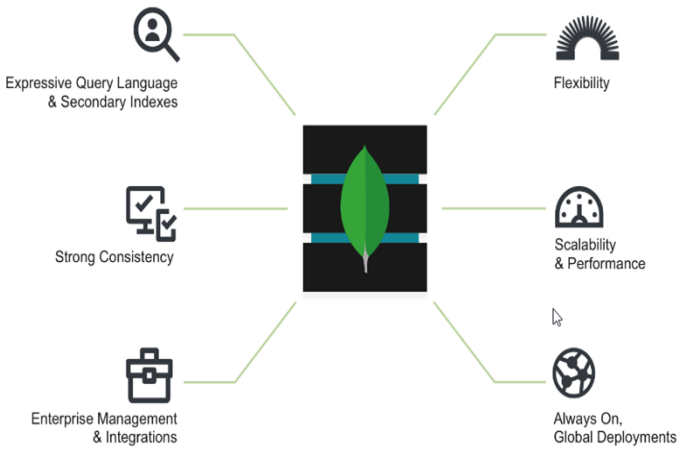

Figure 4-1 MongoDB architecture

MongoDB stores data as documents in a binary representation called BSON (Binary JSON). Documents that share a similar structure are typically organized as collections. Collections can be considered as being analogous to a table in a relational database: documents are similar to rows, and fields are similar to columns. MongoDB documents tend to have all data for a given record in a single document, whereas in a relational database information for a given record is usually spread across many tables. Example of a document structure presented in Figure 4-18

Figure 4-2 MongoDB Document

This database was chosen because MongoDB provides ACID properties at the document level, one or more fields may be written in a single operation including updates to multiple sub-documents and elements of an array. The ACID provided by MongoDB guarantees to ensure complete isolation as a document is updated; any errors cause the operation to roll back so that clients receive a consistent view of the document.

4.2.5. Apache Tomcat

Apache Tomcat [23], often referred to as Tomcat Server, is an open-source Java Servlet Container developed by the Apache Software Foundation (ASF). Tomcat implements several Java EE specifications including Java Servlet, JavaServer Pages (JSP), Java EL, and WebSocket, and provides a "pure Java" HTTP web server environment in which Java code can run. Tomcat is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation, released under the Apache License 2.0 license, and is open-source software. Tomcat flow of execution can be seen in Figure 4.3.

Chosen due to the fact that it can be easily integrated with Java, implements several Java EE specifications and provides a "pure Java" HTTP web server environment.

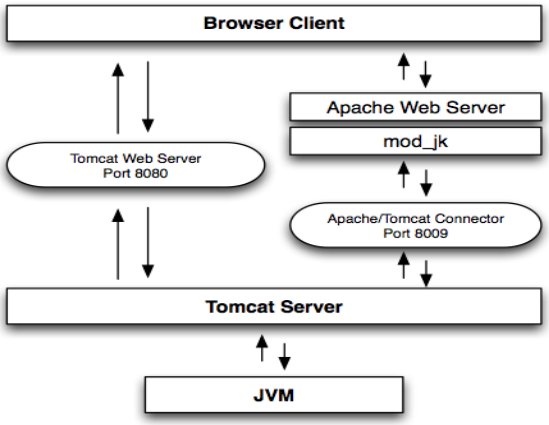

Figure 4-3 Tomcat flow

4.2.6. iText

iText [24] is an open shource libray that can be used to create and manipulate PDF files. Using two classes from this library, it successfully can be created pdf files that contain data from the database or any other source. Additional details can be found in the official documentation, but also from resouorces of type ebook that are available on the developers site that was presented in the previous bibiliographical reference.

4.2.7. Apache POI

Apache POI [25] is a project that has as purpose the creation and maintanence of an API through wich Java can offere support for edting files that are of differenct formats based on the Open Office (OOXML) formats. This library supports the conversion and manipulation of files of type Word necessary when creating the proceeding of the conference. Additional details about the implementation and integration can be found in the official documentation mentioned in the previous bibliographical reference.

4.2.8. Jsoup

Jsoup [26] is a Java library for working with real-worl HTML. It provides avery convinient API for extracting and manipulating data using the best of DOM, CSS, and jQuery-like methods. Additional details about the implementation and integration can be found in the official documentation mentioned in the previous bibliographical reference.

4.2.9. Apache PDFBox

Apache PDFBox [27] is an open-source Java library used for working with PDF documents.This library supports the creation of new PDF documents, manipulation of existing documents and the ability to extract contents from documents. Additional details about the implementation and integration can be found in the official documentation mentioned in the previous bibliographical reference.

4.2.10. Lombock

Lombock [28] is a library that can be used in Java. Is used to avoid repetitive code because Java gets too verbose for things that represent common tasks. It works by plugging into the build process and autogenerating Java bytecode as per number of projet annotations that are introduce in the code. Additional details about the implementation and integration can be found in the official documentation mentioned in the previous bibliographical reference.

4.3. API

4.3.1. Google translate

Google Translate [29] is a free multilingual machine translation service developed by Google, used to translate text, speech, images, sites, or real-time video from one language into another. Additional details about the implementation and integration can be found in the official documentation mentioned in the previous bibliographical reference.

4.3.2. PayPal

PayPal [30] is a service that enables payment, sending money, and accept payments. By registering a credit card or debit card with a PayPal account can execute payments by simply choosing PayPal at checkout, logging into the PayPal account, and confirming the payment. The PayPal account can be used to shop with millions of merchants and sellers around the globe wherever there is a PayPal logo.

4.3.3. CopyLeaks

CopyLeaks [31] is a plagiarism checker for bussines or acadmic. It uses cloud computing, to find copies of the content through the internet. The search enigne used scanes more than 60 trillion pages of the internet and database. Multiple file formats are supported ,e.g doc, pdf, html and more, in any language. Additional details about the implementation and integration can be found in the previous mentioned bibliographical reference.

4.4. System use case

Use cases offer a global perspective over the behavior and functionalities that the system provides. The functional requirements of the system represent possible use case, but not all the functional requirements can be translated to use cases. Requirements like create, update, read and delete do not require presentation under the form of a use case. For a detailed presentation of the use cases that are present in the system the following sections where used.

4.4.1. System actors

Actors present in the system are the following

- Program chair
- Author
- Reviewer
- Visitor

4.4.2. Use cases

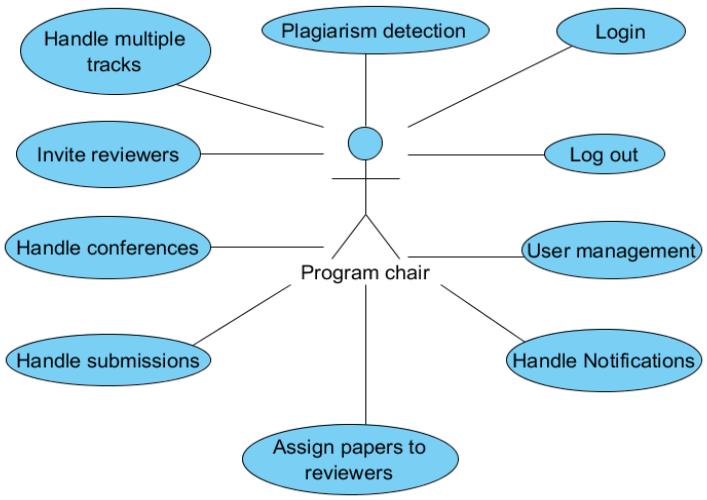

Figure 4-4 Program chair use cases

In Figure 4-4 the available use cases for the program chair are presented.

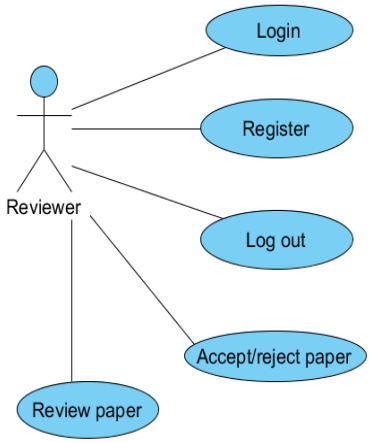

Figure 4-5 Reviewer use cases

In Figure 4-5 use cases for the reviewer are presented. In Figure 4-6 use cases for the author are presented.

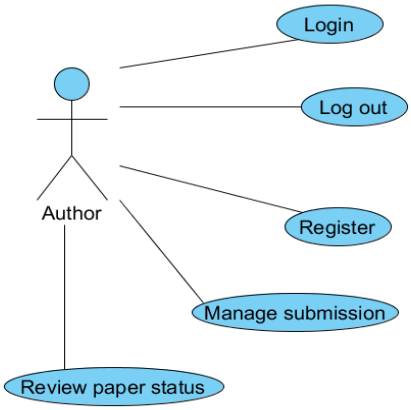

Figure 4-6 Author use cases

Figure 4-7 presents the functionality available for the visitor, functionality that is translated into use cases

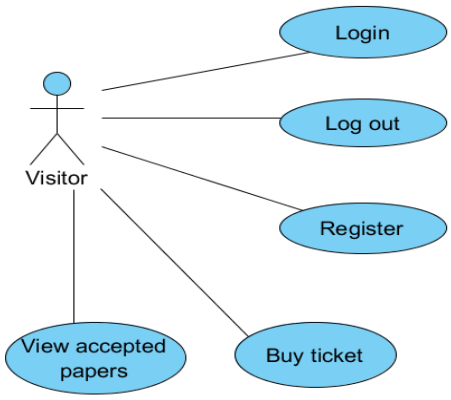

Figure 4-7 Visitor use cases

The next use case presents the execution scenario for the assign of papers to reivewers, repersentative for the functionality available to the admin user.

UC1
Use case name : Assign papers to reviewers
Primary actor : Program chair
Stakeholders:

- **Author** - interested in submitting a paper, by completing a form asking for details about it, in order to obtain the review of qualified people. After the review period has passed if it is positive the paper will be presented to the public
- **Program chair**: interested in having the papers review by qualified peoples
- **Reviewer**: interested in analyzing and review new ideas

Preconditions

- The program chair must be logged in
- An assignment scenario is opened through an Assign Papers operation
- The state of the Conference must be "Review period started" in order to perform a assign. The assignment cannot be performed if the conference is in "Submission are opened" or "Review period not started"

Postconditions

- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create an assignment

1. Program chair proceeds to homepage
2. The system checks the conference timeline
3. The program chair starts an assignment scenario
4. The system checks bag of words computation for the reviewer papers
5. The system checks bag of words computation for the author papers
6. The system checks for similarities
7. The system checks for conflict of interest
8. The paper is assigned to reviewer
9. The user will exit the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies when the review period has not started yet

This flow can occur after step 2
2.1. The use case ends

This alternative flow applies if bag of words for the reviewer has not been computed

This flow can occur after step 4
4.1 The system computes the bag of words
4.2 The use case continues with step 5

This alternative flow applies if bag of words for the author has not been computed

This flow can occur after step 5
5.1 The system computes the bag of words
5.2 The use case continues with step 6

This alternative flow applies if there is a conflict of interest

This flow can occur after step 7
7.1 The system computes the bag of words
7.2 The use case continues with step 6

This alternative flow applies if there are papers that don't have reviewers

This flow can occur after step 9
9.1 The system checks if there are papers that are not assigned
9.2 The use case continues with step

Figure 4-8 presents the flow of events of the above describes use case

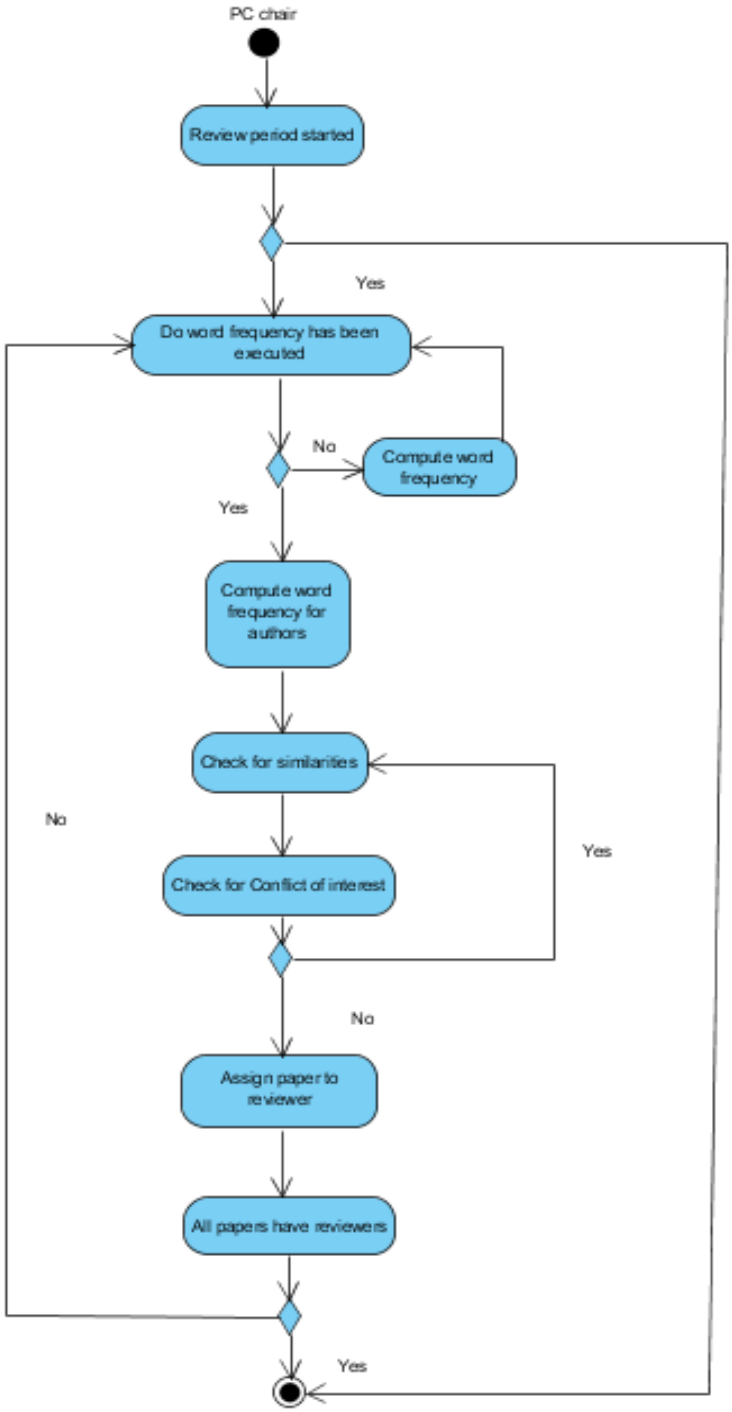

Figure 4-8 Flow chart diagram for assigning papers to reviewers

UC2

Use case name : Add conference
Primary actor : Program chair
Stakeholders:

- **Author -** interested in submitting a paper, to a conference
- **Program chair**: interested in promoting new ideas
- **Reviewer**: interested in analyze and review new ideas

Preconditions

- The author must be logged in and be authorized to perform this use case
- A conference addition scenario is opened through an Add Conference operation

Postconditions

- The conference information is saved in the application database
- Conference entity should remain in a consistent state if the Save/Commit operation has ended with success.
- Conference entity should not suffer any changes if the Save/Commit operation has not ended with success.
- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create/modify a conference

1.Program chair proceeds to the conference submission form
2.The program chair enters the identification data for the conference (e.g. Name, Deadlines)
3.The system checks the data
4.The system will indicate that the conference information has been registered to the system
5.The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies if the system halts

This flow can occur at any step

1. The program chair logs in again in the system, and goes to the add conference page

This alternative flow applies if the data is not correct

This flow can occur after step 3

3.1 The program chair re-enters the information
3.2 The system will validate the information
3.3 The use case continues with step

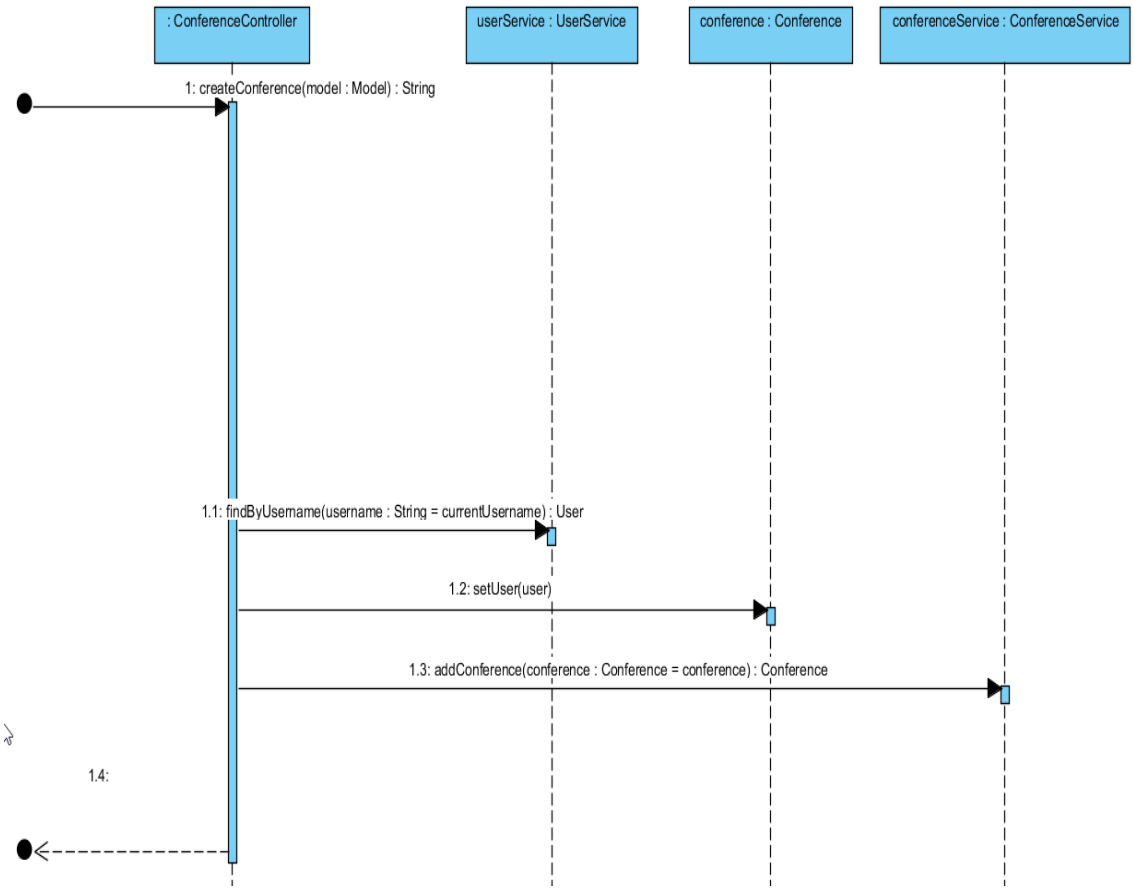

Figure 4-9 Sequence diagram for the add conference use case

The previous sequence diagram, Figure 4-9, presents the communication between the components of the system in order to execute the use case that has as main actor a program chair that wants to create a conference. When accessing the page, the program chair will have available a form that contains a set of fields which need to be filled. A conference consists in a set of fields. Each conference needs to have information about the

conference name, deadlines like conference begin and end, conference submission periods etc.

UC3

Use case name : Inform authors
Primary actor : Program chair
Stakeholders:

- **Author -** interested in submitting a paper, to a conference and get a result
- **Program chair**: interested in promoting new ideas

Preconditions

- The author must be logged in and be authorized to perform this use case
- A handle notification scenario is opened through an Inform authors operation

Postconditions

- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create/modify an entry for the conference

1.Program chair proceeds to the conference home
2.The system sends an email informing the authors about their submissions
3.The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies if the system halts

This flow can occur at any step

1. The program chair logs in again in the system, and goes to the add conference page

This alternative flow applies if the data is not correct

This flow can occur after step 2

2.1. The stmp sends a Delivery Status Notification or a Failure

2.2. The use case ends

The use cases that are presented next are representative for the functionality of the author

UC4

Use case name : Paper submisssion
Primary actor : Author
Stakeholders:

- **Author** - interested in submitting a paper, by completing a form asking for details about it, in order to obtain the review of qualified people. After the review period has passed if it is positive the paper will be presented to the public
- **Program chair**: interested in promoting new ideas
- **Reviewer**: interested in analyze and review new ideas

Preconditions

- The author must be logged in and be authorized to perform this use case
- A submission scenario is opened through an Add Submission operation
- The state of the Conference must be “Submission are opened” or “Submission are available” in order to perform a submission. The submission cannot be performed if the conference is in “Submission are not available” or “Review period started”

Postconditions

- The submission information is saved in the application database
- Conference / Submission entities should remain in a consistent state if the Save/Commit operation has ended with success.
- Conference / Submission entities should not suffer any changes if the Save/Commit operation has not ended with success.
- Project database should not suffer any change if only save (not commit) operations were executed with success on scenario

Basic Flow

Use-Case Start

This use case starts when the actor wants to create/modify an entry for the conference

1.Author proceeds to the conference submission
2.The system checks the conference timeline
3.The author proceeds to the page that contains the submission form
4.The author enters the identification data for the submission (e.g. Name, Keywords)
5.The author selects the physical file he/she wants to upload as paper
6.The system will indicate that the submission has been registered to the system
7.The user checks the status of its submission
8.The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies when the submissions are not opened yet

This flow can occur after step 2
2.1. The use case ends

This alternative flow applies if the paper is not uploaded

This flow can occur after step 5
5.1 The author re-enters the information
5.2 The system will validate the information
5.4 The use case continues with step

UC6

Use case name : Review paper status
Primary actor : Author

Preconditions

- The author must be logged in and be authorized to perform this use case

Postconditions

- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create/modify an entry for the conference

1.Author proceeds to the profile page
2.The author proceeds to the tab that contains the submissions information
3.The system will display the submissions information
4.The user checks the status of its submission
5.The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies if the system halts

This flow can occur at any step

1. The author logs in again in the system, and goes to the profile page

UC7

Use case name : Register
Primary actor : Author

Preconditions

- A registration scenario is opened through a Register operation

Postconditions

- The account information is saved in the application database
- Project database should not suffer any change if only save (not commit) operations were executed with success on scenario

Basic Flow

Use-Case Start

This use case starts when the actor wants to create an account

1. Author proceeds to registration page
2. The author enters the identification data for the account (e.g. Username, Password)
3. The system checks the data
4. The system will indicate that the information has been registered to the system
5. The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies when the passwords do not match

This flow can occur after step 3

3.1 The author re-enters the information
3.2 The system will validate the information
3.3 The use case continues with step

This alternative flow applies if the username already exists

This flow can occur after step 3

3.1 The author re-enters the information
3.2 The system will validate the information
3.3 The use case continues with step

The use cases that are presented next are representative for the functionality of the reviewer

UC8

Use case name : Login
Primary actor : Reviwer

Preconditions

- A login scenario is opened through a Login operation

Postconditions

- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create an account

1. Reviewer proceeds to login page
2. The author enters the identification data for the account (e.g. Username, Password)
3. The system checks the data
4. The user continues with its next step

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies when the password is not correct

This flow can occur after step 3
3.1 The author re-enters the information
3.2 The system will validate the information
3.3 The use case continues with step

This alternative flow applies if the username is not correct

This flow can occur after step 3
3.1 The author re-enters the information
3.2 The system will validate the information
3.3 The use case continues with step

UC9

Use case name : Make review
Primary actor : Reviewer
Stakeholders:

- **Author -** interested in submitting a paper, and having it reviewed by specialized personal
- **Program chair**: interested in promoting new ideas
- **Reviewer**: interested in analyze and review new ideas

Preconditions

- The reviewer must be logged in and be authorized to perform this use case
- A review scenario is opened through a Make review operation
- The state of the Conference must be “Review period started” in order to perform a review. The review cannot be performed if the conference is in “Submission are available”

Postconditions

- The review information is saved in the application database
- Submission / Review entities should remain in a consistent state if the Save/Commit operation has ended with success.
- Submission / Review entities should not suffer any changes if the Save/Commit operation has not ended with success.
- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create a review

1.Reviewer proceeds to the assigned papers page
2.The system checks the conference timeline
3.The reviewer enters the identification data for the review (e.g. Name, Comments)
4.The system check the data
5.The system will indicate that the review has been registered to the system
6.The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies when the reviews are not opened yet

This flow can occur after step 2
2.1. The use case ends

This alternative flow applies if the paper is not uploaded

This flow can occur after step 4
4.1 The reviewer re-enters the information
4.2 The system will validate the information
4.3 The use case continues with step

UC10

Use case name : Reject paper
Primary actor : Reviewer

Stakeholders:

- **Author -** interested in submitting a paper, and having it reviewed by specialized personal
- **Program chair**: interested in promoting new ideas
- **Reviewer**: interested in analyze and review new ideas

Preconditions

- The reviewer must be logged in and be authorized to perform this use case
- A reject scenario is opened through a Reject operation
- The state of the Conference must be "Review period started" in order to perform a review. The review cannot be performed if the conference is in "Submission are available"

Postconditions

- Project database should not suffer any change if only save (not commit) operations were executed with success on scenarios.

Basic Flow

Use-Case Start

This use case starts when the actor wants to create a review

1.Reviewer proceeds to the assigned papers page
2.The system checks the conference timeline
3.The reviewer selects the paper that it does not want to review
4.The system updates paper status
5.The system updates the displayed information
6.The user exits the system

Use-Case End

The user will exit the system

Alternative Flows

This alternative flow applies when the reviews are not opened yet

This flow can occur after step 2

2.1. The use case ends

Chapter 5. Detailed Design and Implementation

This chapter contains the description of the system implementation of the web application. The architecture of the system, deployment diagram, database diagram and structure will be presented and some relevant components will be detailed.

5.1. System architecture

The architecture of the system respects the three tired architecture. Three-tier architecture is a client–server software architecture pattern in which the user interface (presentation), functional process logic ("business rules"), computer data storage and data access are developed and maintained as independent modules, most often on separate platforms. The client interacts with the middle layer through a standard protocol, like API or TCP. The middle-tier interacts with the database also through standard protocols, translating the requests that come from the client into database interrogations and other actions like converting the data from the database into client data.

Apart from the usual advantages of modular software with well-defined interfaces, the three-tier architecture is intended to allow any of the three tiers to be upgraded or replaced independently in response to changes in requirements or technology. For example, a change of operating system in the *presentation tier* would only affect the user interface code.

Typically, the user interface runs on a desktop PC or workstation and uses a standard graphical user interface, functional process logic that may consist of one or more separate modules running on a workstation or application server, and a database server or mainframe that contains the computer data storage logic. The middle tier may be multitiered itself (in which case the overall architecture is called an "*n*-tier architecture").

Presentation logic deals with displaying the information requested by the user in a manner that is easy to use and understand. This level communicates with the client by sending to the browser data. This layer makes available for the user an interface that represents the data that are stored in the database, allows the user to add new information in the database, query the data based on some criteria and update the existing information.

Business logic, middle layer, controller the functionality of the system, by processing data, take decisions and evaluations. This layer makes the connection between the adjacent layers, processing the data that is received from the presentation level and convert it to a form accepted by the data layer. In this layer is present the logic that interconnects the pages from the view, logic necessary for data processing and storage, validation for the data that is received from the presentation layer.

Data layer, consists of the database. The information is stored and extracted from here. This level maintains data neutral and independent of the application server and logic rules, providing data on a separate level, resulting in improvements in performance and scalability. At this level, Create, Update, Read and Update operations are executed. This

level communicates with the business level which makes the data available for it in the form of collections extracted from the database.

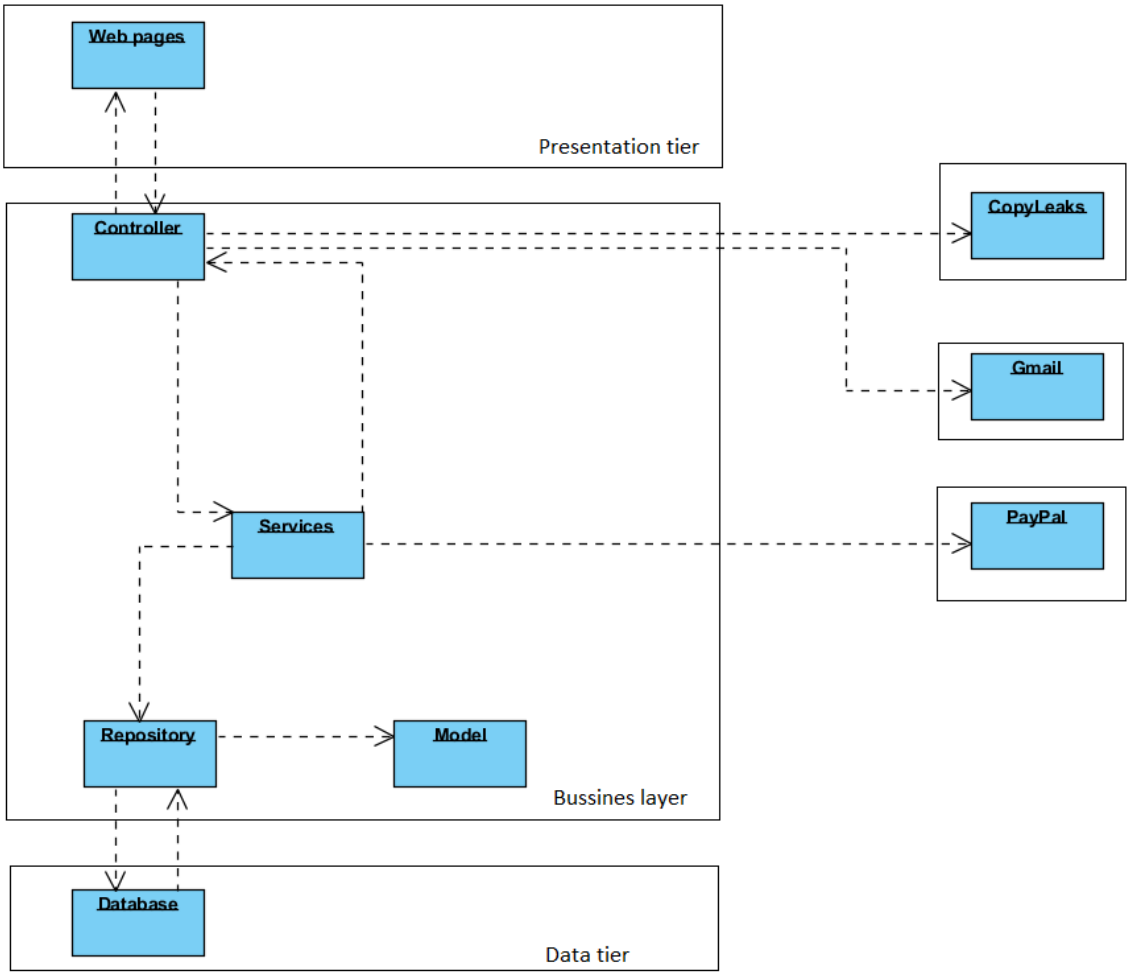

Figure 5-1 System architecture

The application was developed using IntelliJ, development environment, which makes available the development of Spring application in different methods. Method that is chosen by the developer. For the development of this application Spring Boot module was chosen due to the reduction of the configuration time, most of it being provided directly by it. This type of projects contains two important components

- **DemoApplication.java** that contains the main method, method will start the embedded server
- **DemoApplication.jar** present due to the Maven build provided by the Spring Boot Initializer. This jar is handy because it includes all the other dependencies and things like web server inside the archive. This .jar file can be used to run the entire Spring application with no fuss: no build tool required, no setup, no web server configuration

Figure 5-1 presents the general architecture of the system and the connections between the components. Figure 5-2 presents the component diagram of the system. Components that are presented in detail in the following sections

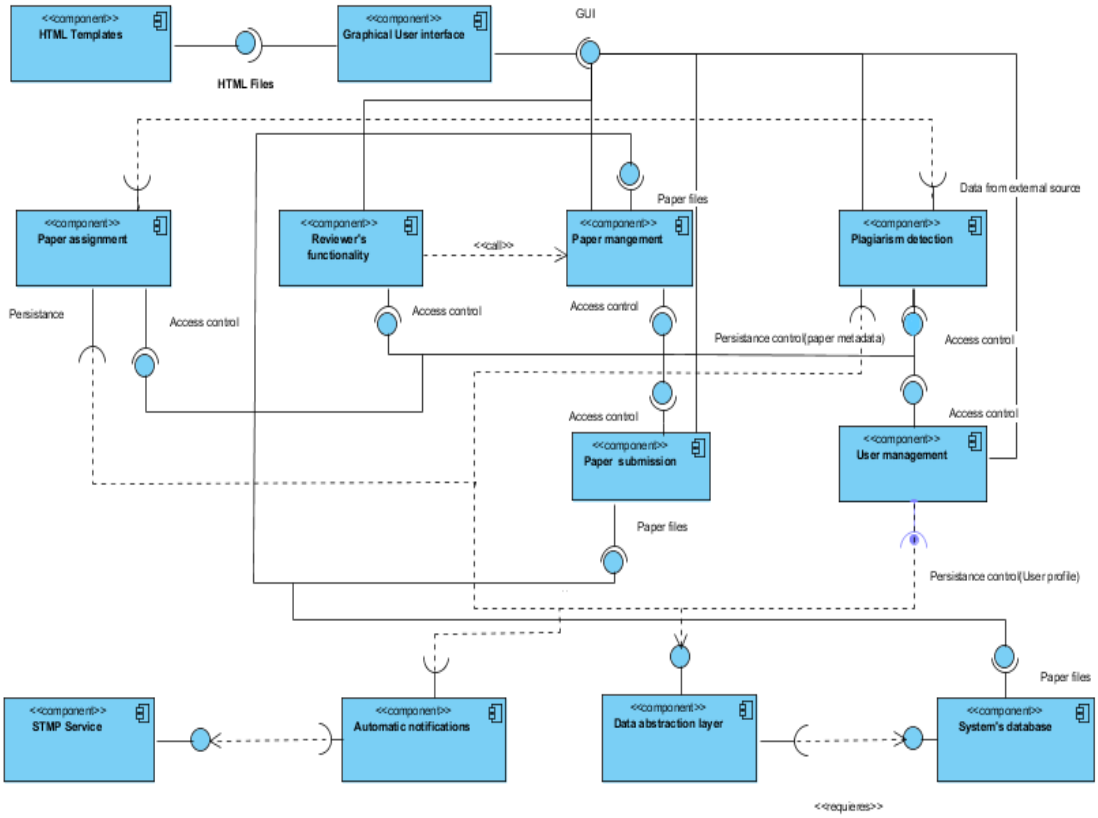

Figure 5-1 Component diagram

5.1.1.1. *Presentation tier*

In this section navigation cases that are common to all users are presented. The web pages where built with HTML language element and Thymeleaf framework, available for Spring Boot, to position the elements. So the logged in user, will have available based on the role one of this pages.

In Figure 5-3 the navigation for web pages common to all roles are presented, In the case of a successful authentication, the user will be redirect to one of the landing pages that are associated to the roles of the application Admin, Author and Reviewer. *About*, *Where* and *updateAccount* pages are available from any page of the application, due to the fact that these pages belong to the header menu.

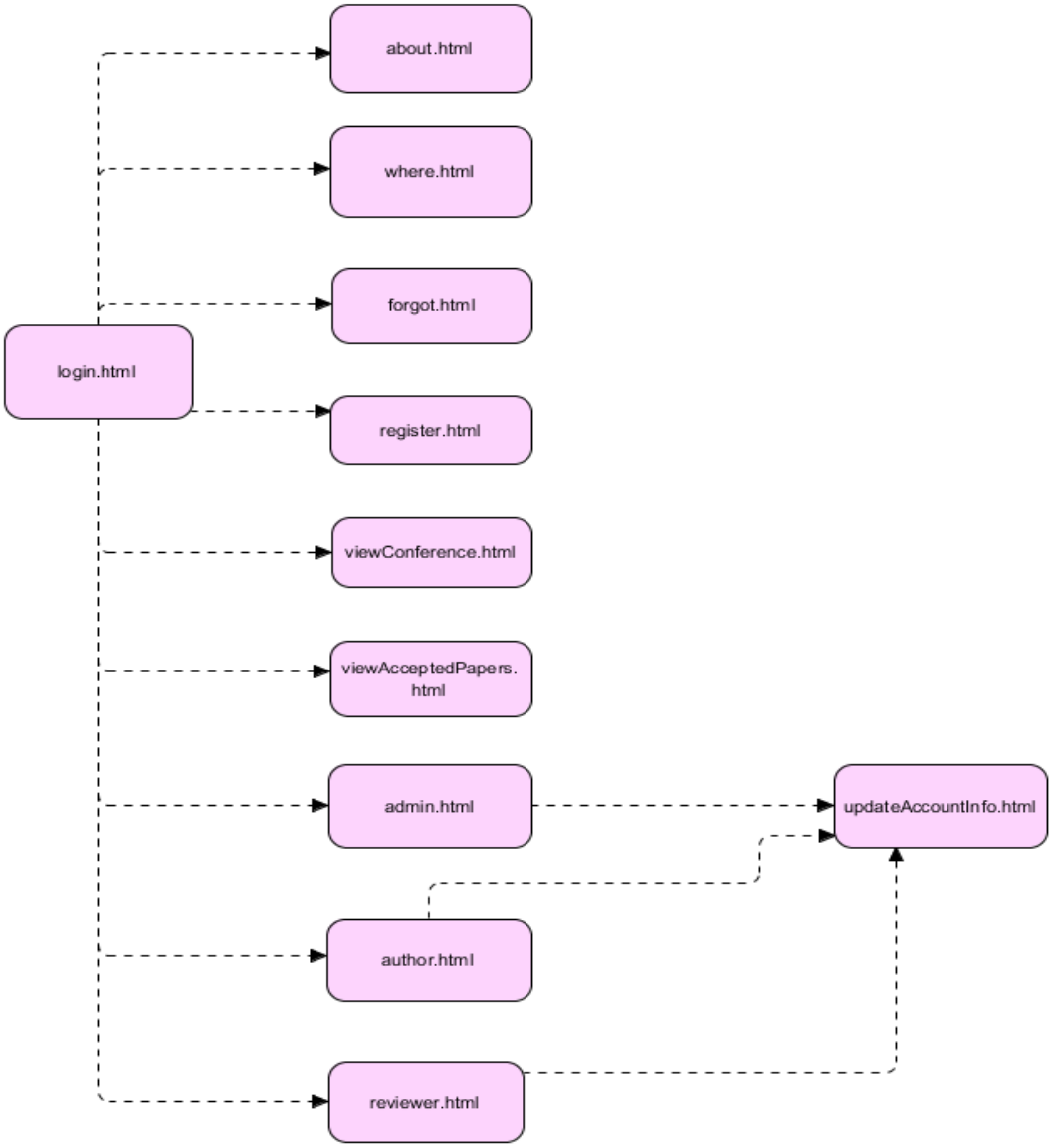

Figure 5-3 Navigation for web pages common to all roles

In the case of an error when trying to login or if the user does not remember his password in order to access the application, there is the option to access the "Forgot Password" from where based on data that the user needs to know (username and email), a notification will be sent to the account associated to the account.

In the next figures the possible navigation routes for the categories of users will be presented.

- **Program chair**

Figure 5-4 presents the possible navigation routes that the user with the administrator role can access after successful authentication

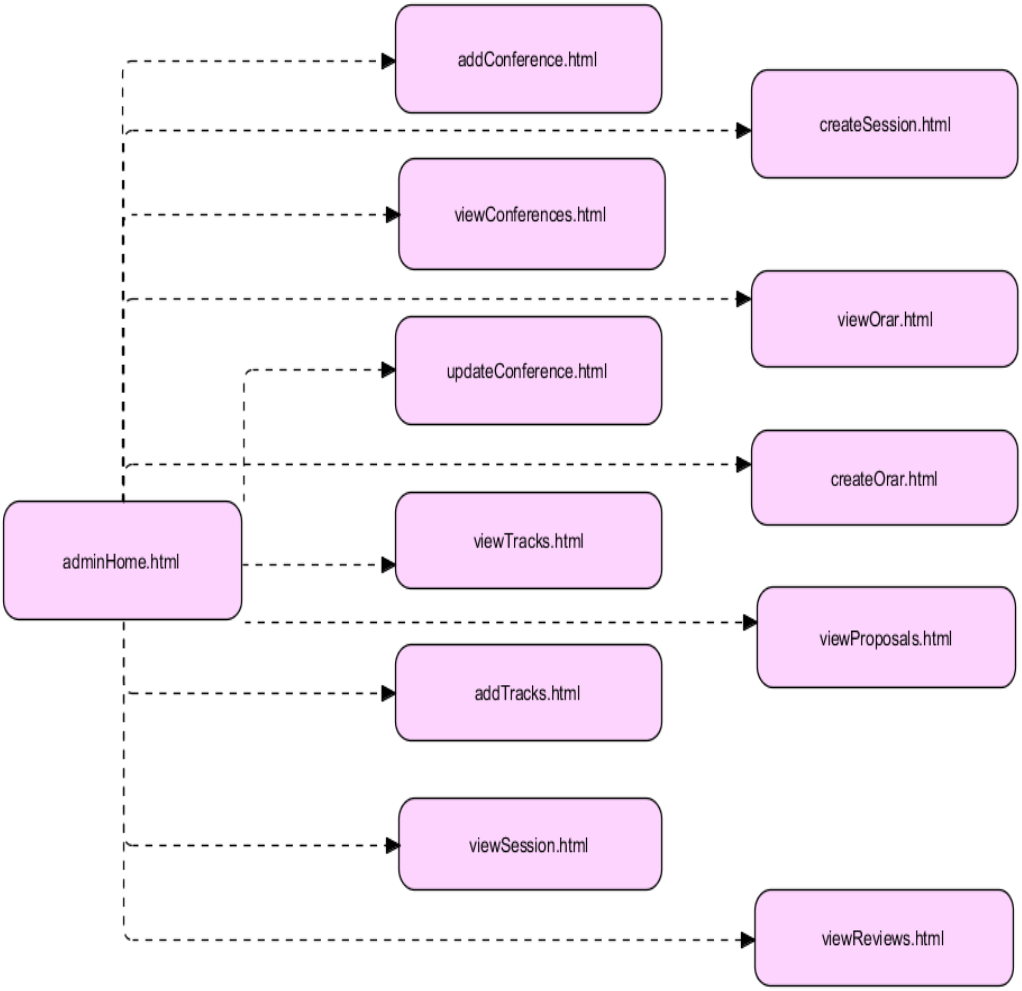

Figure 5-4 Navigation for web pages that are associated to the program chair

- **Author**

Figure 5-5 presents the possible navigation routes that the user with the author role can access after successful authentication.

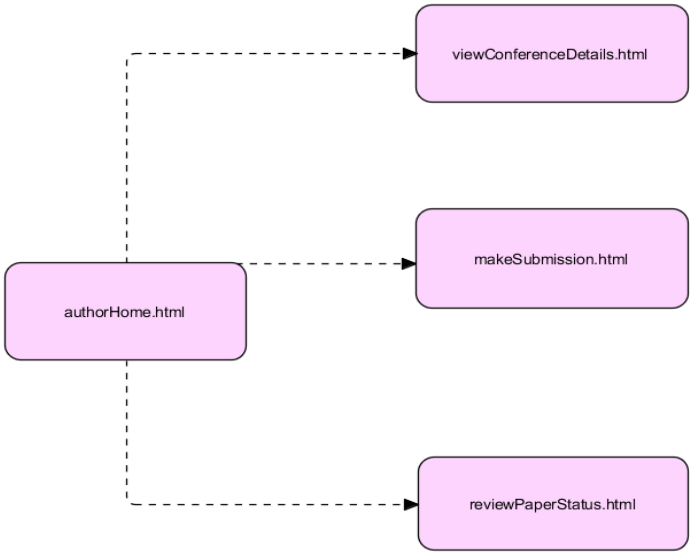

Figure 5-5 Navigation for web pages that are associated to author

- **Reviewer**

Figure 5-6 presents the possible navigation routes that the user with the reviewer role can access after successful authentication

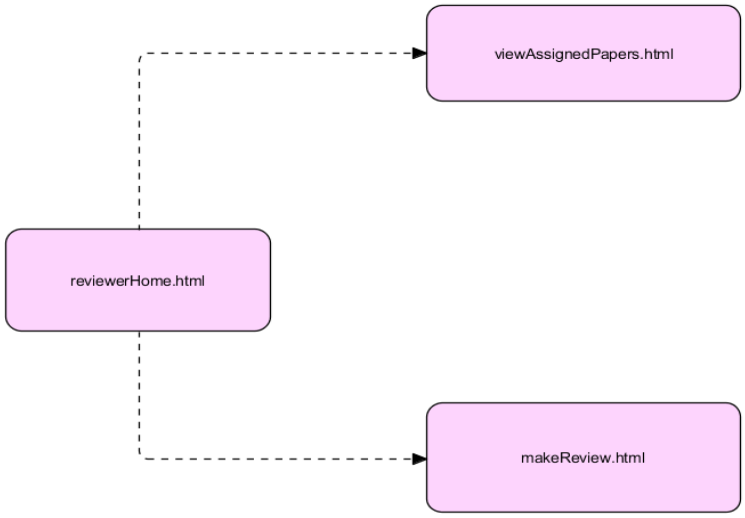

Figure 5-6 Navigation for web pages that are associated to the reviewer

5.1.1.2. *Business Layer*

The next figure, Figure 5-7, presents the packages of the application that contain the logic behind the HTML pages and also are a part of WAR module (web module).

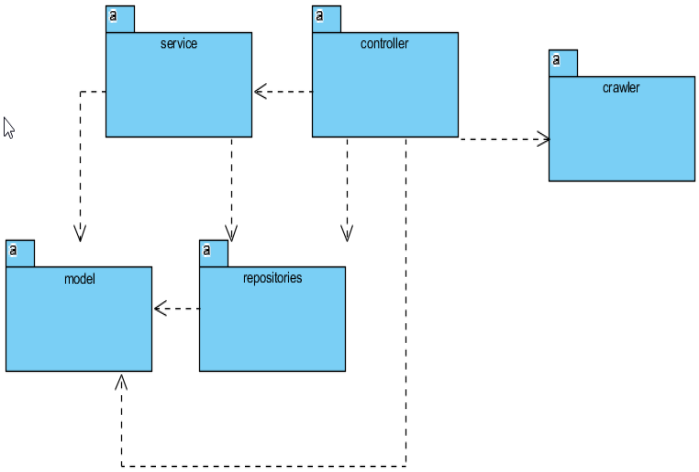

Figure 5-7 Package diagram

1. Controller

Contains the classes that represent the mapping of the requests sent by the dispacher servlet to each process and execute the requested inputs. The URL pattern is important, if the request matches the URL pattern of Dispatcher Servlet then it will be processed by Spring or not. The DispatcherServlet passes the request to a specific controller depending on the URL. The mapping to a specific controller is made using the @RequestMapping annotation of Spring. Controller classes are also identified using @Controller. The class diagram of this package can be seen in Figure 5-8

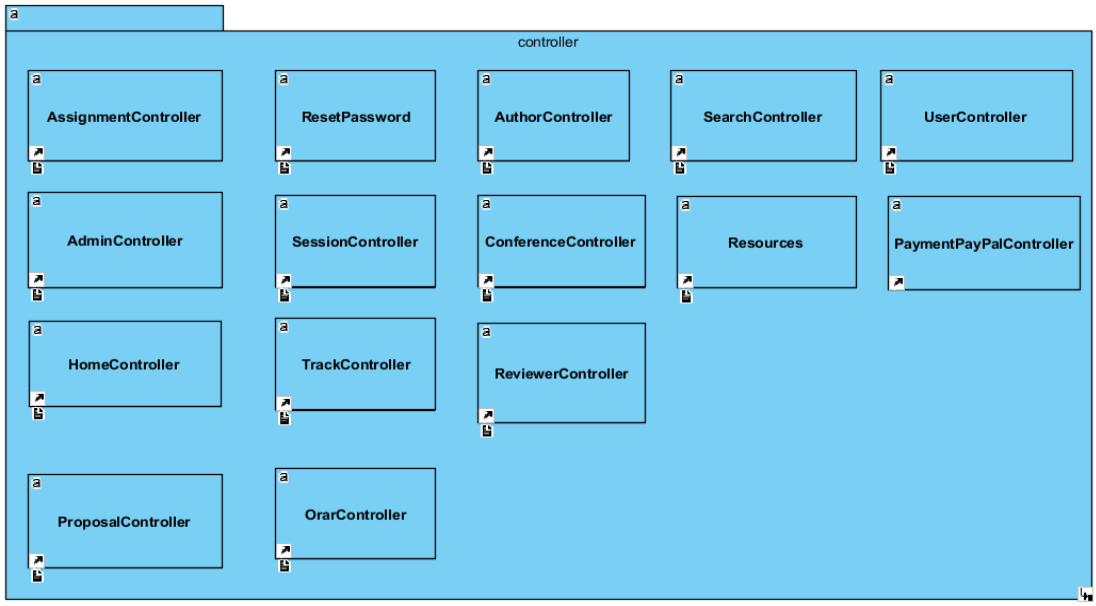

Figure 5-8 Controller class diagram

2. Service

Provides the logic necessary to operate on the data sent to and from the Repository and client. It provides another level of abstraction that would make it difficult to gain access to the database from the client except through the service. Classes that are considered to be services will contain the annotation @Service. The class diagram for this package can be seen in Figure 5-9

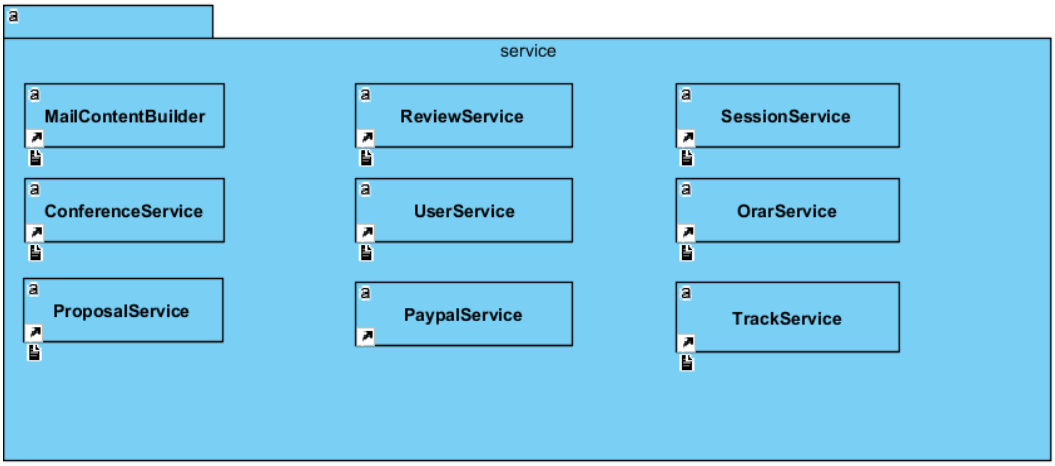

Figure 5-9 Service package class diagram

3. Repository

Provide a specialized implementation of @Component for indicating the Data Access Object (DAO). Advantage of using the @Repository annotation, importing the DAOs into the DI container and also this annotation makes the unchecked exceptions (thrown from DAO methods) eligible for translation into Spring DataAccessException. The class diagram for the repository package can be seen in Figure 5-10

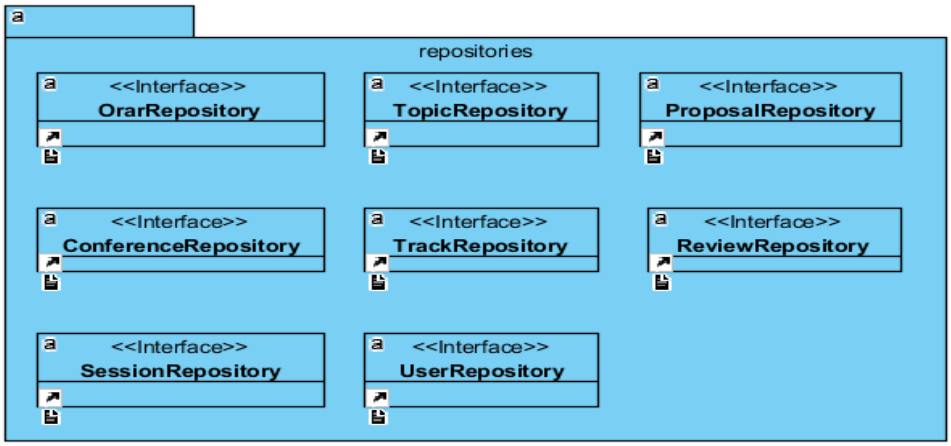

Figure 5-10 Repository package class diagram

4. Model

This component contains the POJO classes. These classes are later mapped to the documents in the database. The relationship between entities will be mapped also in the doocuments present in the database. The next figure , Figure 5-11, presents some of the entities that can be found in the system.

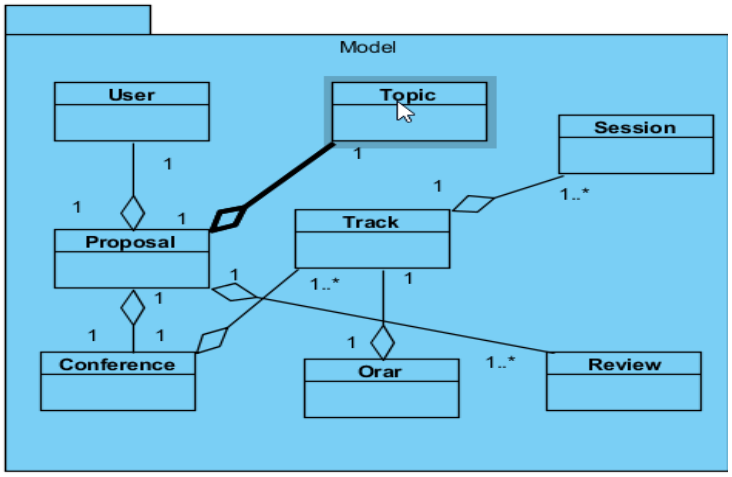

Figure 5-11 Model class diagram

5.2. Deployment

The deployment diagram presents the hardware components on which the software components of the proposed system runs on, but also the way in which they are interconnected. The components of the diagram are also called artifacts of the system. The purpose of this diagram, Figure 5-8, is to present the hardware structure necessary in order to run the system.

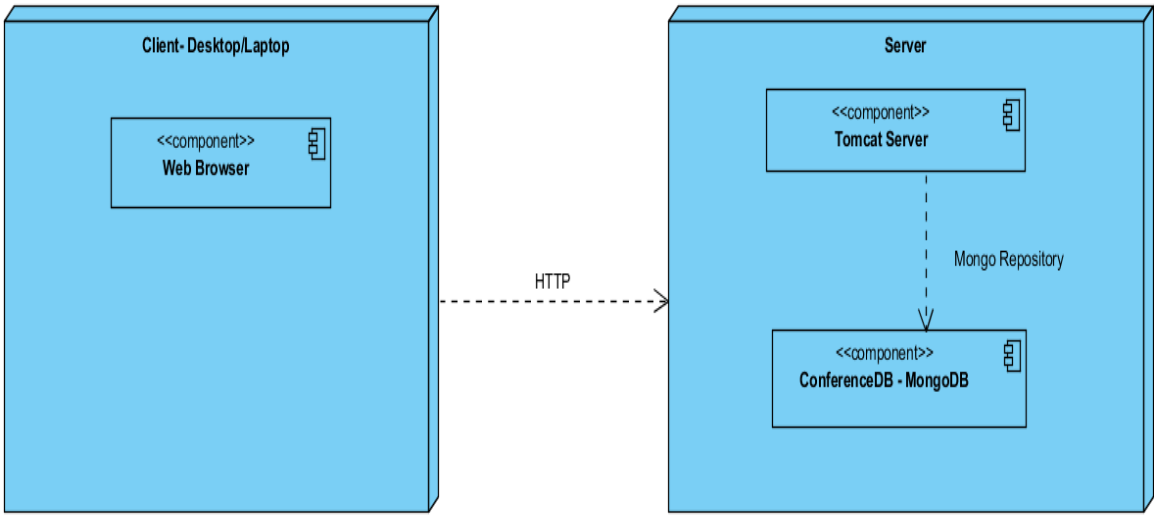

Figure 5-12 Deployment diagram for the system

The web application can also be accessed from any mobile devices, but the development of the graphical user interface was mainly developed for desktop/laptop

access. The client component will communicate with the server component on which the web application is running on through HTTP protocol.

The physical component of the system is the server on which the application runs on, Tomcat server as it can be seen in the right side of the diagram. Also on this component the database which is built in Mongo. On the left hand side of the diagram the Client component is presented, which represent the consumers of the application, the components to which the users have access. The usage of the application can be made from any desktop or mobile end-point, but the only requirement is to have a valid account.

5.3. Important components

5.3.1. Assignment of papers to reviewers

A component that stands out in the web application is the one that deals with the assignment of papers to reviewers and conflict of interest detection. This funcitonality is based on the assignment method proposed in [23] and consists in determining the specialization of a reviewer based on its previous publishing activity. In the graphical user interface, the users with role admin will have available this functionality.

```
for(int i =0; i<papers.size();i++) {
    try {
        PdfReader reader2 = new PdfReader(papers.get(i));
        ArrayList<WordFreq> wordList = new ArrayList<>();
        for(int j=0;j<reader2.getNumberOfPages();j++)
        {
            String text = PdfTextExtractor.getTextFromPage(reader2, j);
            ArrayList<String> words = tokenize(text, stopWords);


            for(int p = 0; p < words.size(); p++){
                if(!wordList.contains(new WordFreq(words.get(p), freq: 1))){
                    wordList.add(new WordFreq(words.get(p), freq: 1));
                }
                else{
                    for(int q = 0; q < wordList.size(); q++){
                        if(wordList.get(q).word.equals(words.get(p))){
                            wordList.get(q).freq++;
                        }
                    }
                }
            }
            for(WordFreq e : wordList){
                System.out.println(e.word + " " + e.freq);
            }
```

Figure 5-13 Word frequency

The implementation of this functionality consists in parsing the pdf files and construct a list of words frequency. The first step is to build a list of word frequency from file that contains a set of words that are considered to be usual, e.g. pronouns, prepositions. This approach was used in order to make a distinction between usual words and the ones that can be used to uniquely identify a paper and the capacity of the reviewer to evaluate the ones submitted to the conference. This functionality can be either extended or improved by using other approaches to determine the qualification of the reviewers. This approach can be seen in Figure 5-13

Another aspect that plays and important role in this component is the conflict of interest detection which is made by comparing the affiliation of the author to the one of the revierwers.

5.3.2. Paper search and download

In order to obtain papers that are used to identify a possible set of reviewers and their capacity to make the best evaluation of the ones submitted to the conference an API to a search engine provide by Google is used. This search engine, Google Scholar, has access to papers that are stored all over the internet.

In order to acces the search engine proxies are used. For a request a proxy is obtained by connecting to a database that stores them. An example of connection can be seen in Figure 5-14. The first parameter represents the URL for the databse, second parameter represents the entity that makes the request and the third parameter represents the time to wait before declaring that the resource is not available

```
doc = Jsoup.connect( url: "http://www.us-proxy.org/").userAgent("Chrome").timeout(5000).get();
```

Figure 5-14 Proxy connection

If the search is successful then the conncetion to the Google Scholar search engine is establish. After this based on the keywords entered by the user the search on the platform is made. The Figure 5-15 shows this approach.

```
keyword = keyword.replace( target: " ", replacement: "+");
//Search google scholar
String url = "https://scholar.google.com/scholar?hl=en&q=" + keyword;
boolean found = false;
while (!found) {
    if (invalidAttempts >= 2) {
        System.out.println("Could not find paper, please try writing more specific information");
        numOfCitations = "";
        citingPapersURL = "";
        found = true;
    } else {
        Document doc;
        try {
            doc = changeIP(url, hasSearchBefore);
        } catch (IOException e) {
            System.out.println("There was a problem connecting to your previously used proxy.\nCh
            doc = changeIP(url, hasSearchBefore: false);
        }


        if (doc.text().contains("Sorry, we can't verify that you're not a robot")) {
            //In case you been flags as a bot even before searching
            System.out.println("Google flagged your IP as a bot.\nChanging to a different one");
            doc = changeIP(url, hasSearchBefore: false);

        }
```

Figure 5-15 Search on google scholar

After the search results are obtained the papers will be downloaded in pdf format. The file is obtained throught the URL address and using the Apache Common library the file is downloaded from that addres, as it can be seen in Figure 5-16

```
private void downloadPDF(String url) throws IOException {
    File docDestFile = new File( pathname: "./DownloadedPDFs/" + pdfCounter + ".pdf");
    URL urlObj = new URL(url);
    FileUtils.copyURLToFile(urlObj, docDestFile);

}
```

Figure 5-16 Paper download

All the possible search results where the paper is cited are obtained using the function presented in Figure 5-17 but in order to avoid the overload of the proxy and to reduce the number of papers downloaded a number of pdfs that will be downloaded is set.

```
private ArrayList<String> getAllLinks() {
    ArrayList<String> list = new ArrayList<>();
    Pattern pattern = Pattern.compile("=\\d*");
    Matcher matcher = pattern.matcher(citingPapersURL);
    String paperID = "";
    if (matcher.find()) {
        paperID = matcher.group();
        paperID = paperID.replace( target: "=", replacement: "");
    }
    //Add 1-10 results
    list.add(citingPapersURL);
    for (int i = 10; i < 1000 + 1; i = i + 10) {
        String sb = "https://scholar.google.com/scholar?start=" + i + "&hl=en&oe=ASCII&as_sdt=5,39&sciodt=0,39&cites=" +
                paperID + "&scipsc=";
        list.add(sb);

    }
    return list;

}
```

Figure 5-17 Get search results

5.3.3. Plagiarism detection

An important component in organzing academic conferences is the need to check for plagiarism for the papers that are submitted. The CopyLeaks API, provides an easy way to load the files to the software. After the file is succesfully loaded the engine provided will chek it agains the components of the database, which contains over 60 trillion pages. An example of usage is presented in the Figure 5-18 which is presented bellow

```
ResultRecord[] results;
CopyleaksProcess createdProcess;
String val = proposal.getPaperPath();
createdProcess = copyleaks.CreateByFile(new File(val), scanOptions);

// Waiting for process completion...
System.out.println("Scanning...");
int percents = 0;
while (percents != 100 && (percents = createdProcess.getCurrentProgress()) <= 100)
{
    System.out.print("\r" + DisplayBar(percents) + " " + percents + "%");

    if (percents != 100)
        Thread.sleep( millis: 5000);
}
System.out.println();
```

Figure 5-18 CopyLeaks usage example

In order to execute the previous presented function a connection to the application needs to be established first. In order to establish the connection a valid account and an API key is necessary. Also for this type of application a valid number of credits is necessary. An example can be seen in Figure 5-19

```
CopyleaksCloud copyleaks = new CopyleaksCloud(eProduct.Businesses);
System.out.print("Login to Copyleaks cloud...");
copyleaks.Login( email: "alyna_943@yahoo.com",  APIKey: "2CCFA91C-0AE8-4F0C-923A-950EB546A9AF"
System.out.println("Done!");

System.out.print("Checking account balance...");
int creditsBalance = copyleaks.getCredits();
System.out.println("Done (" + creditsBalance + " credits)!");
if (creditsBalance == 0)
{
    System.out.println(
            "ERROR: You have insufficient credits for scanning content! (current credits
                    + creditsBalance + ")");
    return "redirect:/admin/viewSubmissions";
}

ProcessOptions scanOptions = new ProcessOptions();
```

Figure 5-19 CopyLeaks connect

5.3.4. Payment component

Another component that will be described is the component that deals with payment. These aspect is dealt using PayPal. In order to have access to this component a valid account with all the necessary permission, then create an application through which a client id and client secret is obtained. An example can be seen in Figure 5-20 which is presented next.

```
paypal.mode=sandbox
paypal.client.id=AXayPhmgOiWjZ_c79acXaeuelO7j6B8HUMuKn1LTx9xVdwL08hdBFGNCGNwmI1egQCwX7VU9ZfQx17jz
paypal.client.secret=ELsQrtBc6ZY3JoIMqX3-kfqN-jZbTCEQmIvmE9ZU8EaCuoCxBPFlrlWJ_MDosiv1jxSmjSpU_0t-ETeI
```

Figure 5-20 PayPal credentials

After these credentials are obtained is necessary to build the request that will process the necessary functionality. An example can be seen in Figure 5-21

```
public String pay(HttpServletRequest request){
    String cancelUrl = URLUtills.getBaseURl(request) + "/" + PAYPAL_CANCEL_URL;
    String successUrl = URLUtills.getBaseURl(request) + "/" + PAYPAL_SUCCESS_URL;
    try {
        Payment payment = paypalService.createPayment(
                total: 15.00,
                currency: "USD",
                PaypalPaymentMeth.paypal,
                PaypalPaymentIntention.sale,
                description: "payment description",
                cancelUrl,
                successUrl);


        for(Links links : payment.getLinks()){
            if(links.getRel().equals("approval_url")){
                return "redirect:" + links.getHref();
            }
        }
```

Figure 5-21 Process payment

5.3.5. Booklet

Another functionality that needs to be presented is how a booklet is created. A booklet for a conference presents information like sponsors, informations about the organizing committeea and informations regarding the submissions that where accepted to the conference like paper abstract and author name. The pdf representing the booklet is created using the library ApachePDFBox and functions provided by it.

5.3.6. Data vulnerability

Access to data for this application is made using the Spring Security component. In oder to **retrieve the currently authenticated principal** the process consists in a static call to the *SecurityContextHolder*. An example can be seen in Figure 5-22

```
Authentication authentication = SecurityContextHolder.getContext().getAuthentication();
String currentUsername = authentication.getName();
```

Figure 5-22 Principal retrieval

One of the main form of attacks that can affect a page built with Thymeleaf and Spring Security is Cross-Site Request Forgery CSRF. The protection against these type of attacks is activated from the start when using Spring Security. Finally, with CSRF protection activated on the server side on the client side additional parameters need to the included in the request. An example can be seen in Figure 5-23

```
<meta th:name="_csrf" th:content="${_csrf.token}"/>
<meta th:name="_csrf_header" th:content="${_csrf.headerName}"/>
```

Figure 5-23 Input request

5.4. Database

The implemented system uses a database built on a platform offer by MongoDB, database that contains a set of 8 collections. In the next diagrams these collections will be presented on categories, based on their importance. Each collection contains a series of documents, where each document maps to a model from the real world.

Normalization is the process of organizing data in a database, according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency, Redundant data wastes disk space and creates maintenance problems. If data that exists in more than one place must be changed, the data must be changed in exactly the same way in all locations.

MongoDB does not support joins. In MongoDB some data is *denormalized*, or stored with related data in documents to remove the need for joins. However, in some cases it makes sense to store related information in separate documents, typically in different collections or databases. MongoDB applications use one of two methods for relating documents:

Manual references where the _id field of one document in another document is saved as a reference. Then in the application a second query to return the related data can be executed. These references are simple and sufficient for most use cases.

DBRefs are references from one document to another using the value of the first document's _id field, collection name, and, optionally, its database name. By including these names, DBRefs allow documents located in multiple collections to be more easily linked with documents from a single collection.

To resolve DBRefs, must be performed in the application additional queries to return the referenced documents. Many drivers have helper methods that form the query for the DBRef automatically. The drivers do not *automatically* resolve DBRefs into documents. DBRefs provide a common format and type to represent relationships among documents. The DBRef format also provides common semantics for representing links between documents if the database must interact with multiple frameworks and tools [22]. In the figure next, Figure 5-24 the database diagram is presented

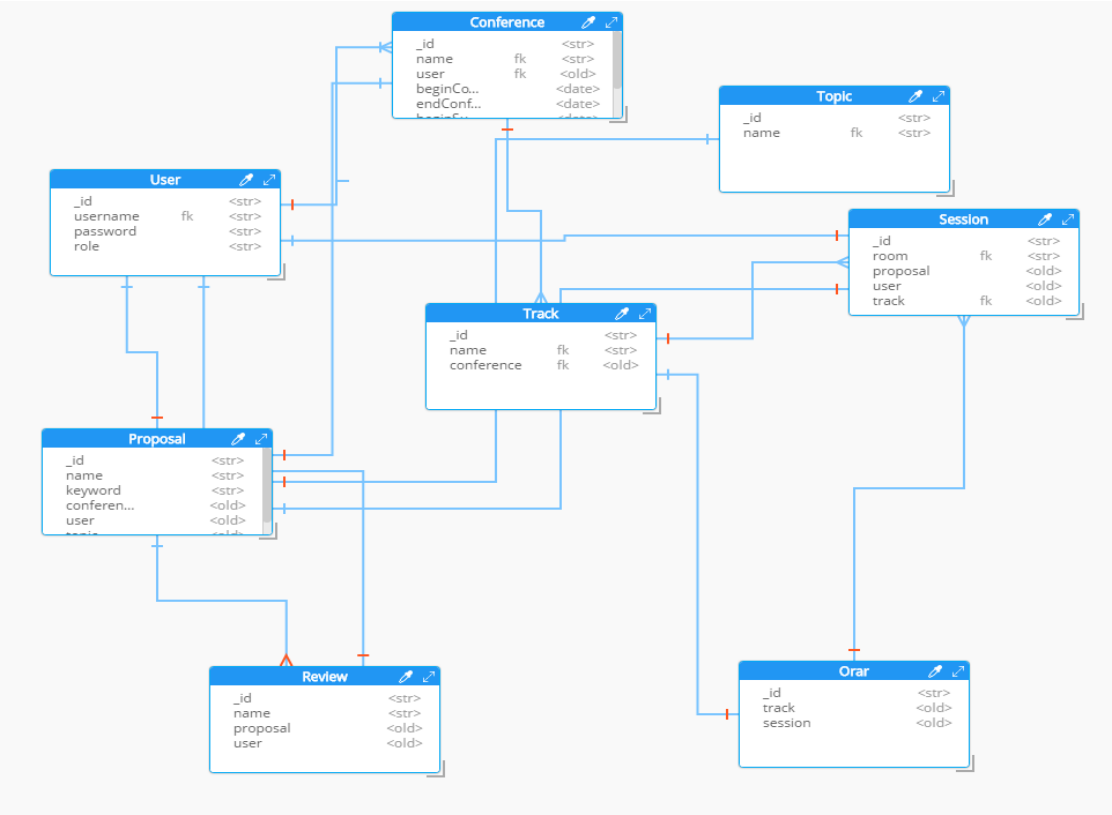

Figure 5-24 Database diagram for the system

For a better management of the documents presented in the database, each document has an _id field generated when inserted in the database that uniquely identifies it. The relationships that are presented in the database are one to many and one to one. Between the relationships that are presented in the diagram above the most important are the one to many between the proposal and the conference, one to one between the proposal and the user.

Between the Conference collection and the Track collection is a one to many relationship taking into consideration that one conference can have more than one track, but a track can have associated only one conference.

Between the track and session collections there exists a one to many relationship taking into consideration the fact that one track can have more than one session, but a session can belong to only one track. Having a closer look to the Session and Paper collections there is a one to one relationship considering that a Session can have only one Proposal, and the fact that a Proposal can be presented in only on session.

Considering the collection Session and User there exists a one to one relationship between these two due to the fact that a Session can have only one user that presents its paper, and a User can present in only one session his proposal. Taking into consideration that one of the fields used in the has additional data, in the previous mentioned diagram the short description was presented.

The document User contains all the data necessary for all users, which will be divided into categories based on roles

- Username – used for the authentication in the database
- Password
- First Name
- Last Name
- Email – used to receive emails
- Role – these field represents the authority

For each submission made there will be associated the user that makes the submission and conference information. The schedule is built for each track, so there will be a one to one relationship between the track and the schedule. This functionality is managed by the administrator.

Chapter 6. Testing and Validation

In this chapter the main testing processes that were used for these system is presented. Testing in general does not guarantee the complete and correct functioning in all conditions of the system, but can be helpful in order to identify the problems and maybe provide some solutions. Testing process can be defined as the process of validation and verification that the system satisfies the imposed requirements and constraints.

The development process was an iterative one, which required testing of each component or the functionality when the necessary implementation was finished. The testing was realized manually, starting from simple success scenarios, to entering wrong data and observe the states that the system reaches.

Besides the manual testing a suite of unit tests and integration tests where done in order to verify the functionality of the system. Unit testing involves breaking the program into pieces, and subjecting each piece to a series of tests, as it can be seen in the next figures.

```
@Test
public void insert() {
    List<User> users = userRepository.findAll();
    int intitialSize = users.size();
    User project = new User( username: "alinaTes");
    userRepository.save(project);
    List<User> users2 = userRepository.findAll();
    int afterIns= users2.size();
    Assert.assertNotEquals(intitialSize,afterIns);
}
```

Figure 6-1 Unit test

The @Test annotation tells Junit that the method to which is attached can be run as a test case. In order to run the method first Junit will make an instance of the class then invokes the annotated method. Any exceptions thrown by the test will be reported by Junit as a failure. An example can be seen in Figure 6-1. A set of result can be seen in Figure 6-2. This tests were conducted using data that will led to a positive result and a suite of data that will result in a failed test.

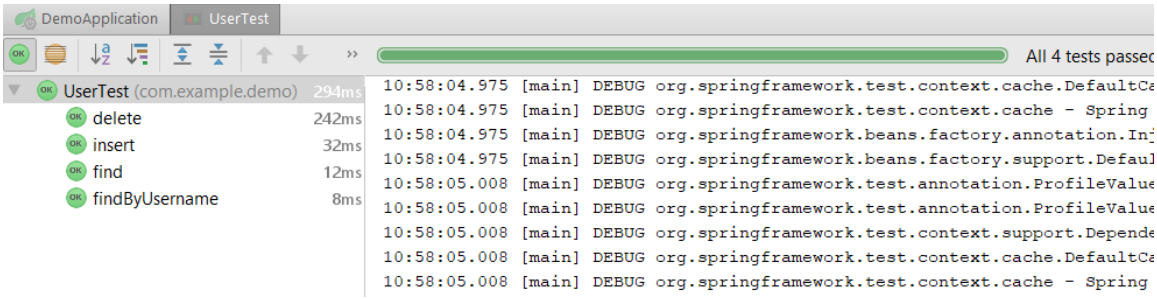

Figure 6-2 Test results

Another suite of tests that where conducted are integration tests. An integration test is done in order to demonstrate that a piece of the system works in a correct manner. Integration tests cover the whole application. They require resources like database instances and hardware allocated to it. In order to execute this category of tests, an additional library was used, Mock, that can simulate the behavior of a service. An example can be seen in Figure 6-3

```
try {
    when(userService.findAll()).thenReturn(userList);

    System.out.print(userList.size());
    assertEquals( expected: 2,userList.size());

    for(int i=0;i<userList.size();i++)
    {
        System.out.println(userList.get(i).getUsername());
    }

    assertTrue(user.getUsername().equals(userList.get(0).getUsername()));
    assertTrue(user2.getUsername().equals(userList.get(1).getUsername()));
} catch (Exception e) {
    e.printStackTrace();
}
```

Figure 6-3 Integration test

Chapter 7. User's manual

This chapter contains wrapped in a tutorial the necessary steps to install the components of the system and maybe followed by a deployment on a local machine. Also in this chapter the necessary hardware and software resources, and also a short user manual.

7.1. Necessary dependencies

For this web application, specific hardware resources are not necessary only the resources necessary to run a browser. The deployment of the application can be done in the local network which will allow access to all the computers connected.

The necessary dependencies for the applications are the following

- Java version 1.8
- IntelliJ IDEA version 2017.1
- MongoDB version 3.4.6

7.1.1. Install and run

Next steps are presented in order to offer the possibility to create a new instance of the application. The operating system that is recommended in Windows, due to the fact that a person without technical knowledge can execute the following tutorial.

- **Installing IntelliJ IDEA**

In order to install the application in a short period of time so that Personal Computer plays the server role, it is necessary to download and install the Java Development Kit and the environment IntelliJ IDEA which can be found at [31]. After download has finished successfully, the download file represents an executable which only requires to choose the installation path

- **Tomcat server**

The application is built using Spring Boot, the additional installation of Tomcat is not necessary because Spring Boot provides an emended one that is automatically configured

After this was successfully installed and configure the next step is to download MongoDB and start it

- **MongoDB**

Installing MongoDB is a similar process to installing IntelliJ, it is necessary to access the site at the address [32]. After the download has completed successfully, the obtained file is an executable and the installation requirements can be found on the developer's page.

- Installing MongoDB is an iterative process that is easy to understand and execute with the instruction that can be found on the developer's web page

- When the install has finished. The establishment of the connection to the server allows the creation of the **ConferenceMang** database and the collection that are part of it will automatically be created when new data will be inserted
- In order to configure the access to the database some parameters must be specified. And after the access it will be automatically configured by Spring data. How parameters are specified it can be seen in Figure 7-1

```
@Configuration
@EnableMongoRepositories(basePackages = "com.example.demo.repositories")
public class MongoConfiguration extends AbstractMongoConfiguration {
    @Override
    protected String getDatabaseName() {
        return "conferenceDatabase";
    }

    @Override
    public Mongo mongo() throws Exception {
        Mongo client = new MongoClient( host: "localhost", port: 27017);
        return client;
    }


    @Bean
    public MongoTemplate mongoTemplate() throws Exception {
        return new MongoTemplate(mongo(), getDatabaseName());
    }
}
```

Figure 7-1 MongoDB Configuration

- **Project import**

In order to import the project and start using it is necessary to open IntelliJ, select the menu File->New project->Project from existing source and then select the path to the project source.

- **Project run**

The last and final step in order to have the application functioning is to run it. This step consists in selecting the class **DemoApplication** and finding the method *main* which will deploy the application on the server, and the application will be available by navigating to the address *"http://localhost:8080"* which will display the view of the application

7.1.2. User manual

In order for the authentication to be successful, the users first need to have a valid account. If the account exists, the user has to enter the username and password in the fields present in the login page, otherwise register. In the Figure 7-3 is presented the login page. After the authentication is executed successfully the user can start using the application.

Figure 7-3 The login form for the system

In the Figure 7-4 the graphical user interface for the user with the role Program Chair is presented which after the successful login, can start using the application

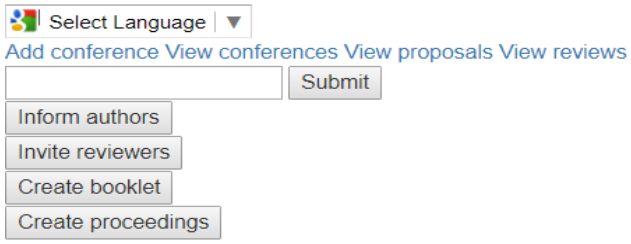

Figure 7-4 Graphical user interface for the program chair

The previous figure illustrates the graphical user interface that the user with the role program chair has available. As it can be seen in the figure the interface contains a set of buttons, a search box and some additional functionality.

The user will have available at any point when using the application an escape plan, a possibility to leave the application, to return to the home page, usually this functionality is implemented through a logo, that is also present for the current application.

Chapter 8. Conclusions

In this chapter the functionality that was implemented, the objectives that were fulfilled through this project and also possible further developments and improvements are detailed in the following sections.

8.1. Objectives met

The system that was developed has met its objectives, to be a system that can compete with the similar ones that are available.

The proposed secondary objectives were fully implemented, the users are classified on roles, conferences can be created, deadlines are managed, submissions can be made, managed and reviewed. The assignment is made in the automatic manner same as detection of conflict of interest. Invitation of reviewers is made through the mail service

The component that deals with finding and downloading papers on the storage platform provide by google was tested on a small set of data, but this can be improved, extended further as the system grows.

The component that deals with the assignment of papers to reviewers was tested on a small set of data, but the obtained results can be improved as the system gets more learning data or other ways for executing this functionality in a manner that provides a better performance.

8.2. Further developments

Usually any system can be further improved, improvements of any nature, from adding new functionality to improving and perfecting the existing ones, and the developed system does not represent an exception.

- Taking into consideration that the number of users that have intelligent phones which can use different applications a possible development would be the implementation of a mobile application initially for Android
- Extend to operating systems like iOS and Windows phone
- Adding new functionality that helps users make a reservation for hotels, restaurants that are near by
- Add functionality that allows the program committee to organize technical workshops
- Add functionality that allows participants to choose to which session to participate when they buy a ticket
- Another possible development would be adding new roles to the system, like technical committee, program sub-chair or speaker. Each of these roles would have specific functionality. Users with role of sub-chair would take part of the administrator's responsibilities in order to reduce the workload. Users that would be part of the technical committee that deals with problem that can appear in the chosen domain. Users with role of speaker are selected to raise interest in a particular domain

- Possibility to create templates that are dynamic for sending emails.
- Possibility to create dynamic templates for reviews based on the domain chosen by the program chair
- Possibility to create a review form for the participants in order to evaluate their experience, and based on that evaluation to improve the points suggested by the participants.

Bibliography

[1] D. Deniz, A. Bulancak**,** *WCMS: Web-Based Conference Management System,* Information Technologies R&D Center, Department of Electrical & Electronic Engineering, Eastern Mediterranean University, available at http://ace.ucv.ro/sintes13/SINTES13_2007/Invited%20sesion/SI%20-%2004%20-%20Dervis%20Deniz.pdf

[2] F. Azzeh, H. Mimi, A. Aldahoud, K. Awad**,** *CMAS: An Online Conference Management and Archiving System*, Alzaytoonah University of Jordan, Jordan, available at http://www.ubicc.org/files/pdf/695_695.pdf

[3] Y. Kalmukov, "Architecture of a Conference Management System Providing Advanced Paper Assignment Features"*, International Journal of Computer Applications* (0975 – 8887) Volume 34– No.3, November 2011, DOI: 10.5120/4083-5888

[4] K. Daimi, L. Li, *Designing an Online Conference Management System*, Department of Mathematics, Computer Science and Software Engineering, University of Detroit Mercy,4001 McNichols Road, Detroit, MI 48221, available at http://worldcomp-proceedings.com/proc/p2011/SER3957.pdf

[5] K. Ahmad, A. Abdullah and A. Zeki, "Web-based Conference Management System for Higher Learning Institutions*", 2012 International Conference on Advanced Computer Science Applications and Technologies*, **DOI:** 10.1109/ACSAT.2012.22

[6] EasyChair system, http://easychair.org/

[7] ConfTool system. http://www.conftool.net/index.html

[8] Microsoft system, https://cmt.research.microsoft.com/cmt/

[9] P. Rigaux, *The MyReview System, https://www.its.hku.hk/news/ccnews132/myreview.htm*

[10] M. Jain, T. Tewari and S. Singh, "Survey of Conference Management Systems", *International Journal of Computer Applications* (0975 – 8887) Volume 2 – No.2, May 2010, DOI: 10.5120/632-875

[11] O. Pop, *Information Systems lectures,* Technical University of Cluj-Napoca, 4th year course

[12] Java, http://www.oracle.com/technetwork/java

[13] Maven, https://maven.apache.org/

[14] Spring framework reference documentation, http://docs.spring.io/spring

[15] Spring boot reference documentation, http://docs.spring.io/spring-boot/

[16] Spring security reference documentation, https://docs.spring.io/spring-security/

[17] Salted password hashing tutorial, https://crackstation.net/hashing-security.htm

[18] Spring data MongoDB, http://docs.spring.io/spring-data/mongodb/

[19] Spring Thymeleaf reference documentation, http://www.thymeleaf.org

[21] NoSQL tutorial, http://basho.com/resources/nosql-databases/

[22] MongoDB, https://www.mongodb.com/resource-center

[23] Tomcat server, https://tomcat.apache.org/

[24] iText official documentation, http://itextpdf.com/learn

[25] Apache POI, http://poi.apache.org

[26] Jsoup, https://jsoup.org/
[27] Apache PDFBox, https://pdfbox.apache.org/
[28] Lombock, https://projectlombok.org/
[29] Google Translate, https://translate.google.com/manager/website/?hl=en
[30] PayPal, https://developer.paypal.com/developer/
[31] L. Charlin, R. Zemel, *The Toronto Paper Matching System: An automated paper-reviewer assignment system*, University of Toronto, available at http://www.cs.toronto.edu/~lcharlin/papers/tpms.pdf
[32] IntelliJ IDEA official site, https://www.jetbrains.com/

Appendix 1 List of figures

Table list from the paper

Appendix 2 Glossary

API	Application programming interface
Hashing function	Function defined on an infinite set of data with values that are a part of a finite set and a finite set of numbers
HTTP	Hypertext Transfer Protocol
Java SE	Java Standard Edition
Java EE	Java Enterprise Edition
JDK	Java Development Kit
POJO	Plain Old Java Object
Salt	In cryptography, salt represents a set of random data used as additional data to a one way hashing function for the password
TCP	Transfer Control Protocol
URI	Uniform resource locator
UUID	Universally Unique Identifier