**Conference Management System**

**1. Introduction**

The process of managing informaton required for organizing a conference is a time-consuming activity. In order to produce and manage this information, using a dedicated software application is from far the best solution.

**2. Current System**

There isn't any already existing system on which the proposed system is build. The system is build from scratch

**3. Proposed System**

**3.1 Overview**

The system ensured that users of every role (chair, reviewer, author) can obtain all the information required for their need with respect to the conferences in which they are engaged. A distinction is made between the users of the three roles. There are functionalities common to all three roles, as well as functionalities specific to every role. Beside access to conference information, the users can also add information to the system. A chair can modify the details of the conference for which they are responsible. An author can submit papers to a conference. A reviewer can label a paper submitted by an author as accepted or rejected.

**3.2 Functional requirements**

a. Common functionalities:

* Users (chair, author, reviewer) sign in using their unique username or email, and their password.
* POST: url + /user/login
* Register: POST: /user/register
* Users can see their personal information (first name, last name, date of birth, phone number, address). X
* GET: url + /user/{id of the user}
* Users can update their personal information. X
* PUT: url + /user/
* Users can see the list of all conferences. X
* GET: url + /conference/
* Users can see the details of any conference. X
* GET: url + /conference/{id of the conference}
* User can see all the four deadlines of a conference. X
* Same as above
* Users can see all the topics of interest of a conference. X
* Same as above

b. Chair functionalities:

* A chair can update the conference details of any conference they preside over. X
* PUT: /conference/{id of the conference}
* A chair can update the topics of interest of any conference they preside over. X
* Same as above
* A chair can see all the papers in any conference they preside over. X
* GET: /conference/{id of the conference}/papers
* A chair can make the final decision regarding the status of any paper (accepted/rejected) in any conference they preside over. X
* PUT: /conference/{id of the conference}/papers/{id of the paper}/decideOnPaper
* A chair can see all the accepted papers in any conference they preside over. X
* GET: /conference/{id of the conference}/papers
* A chair can assign any accepted paper to any of the conference sessions of any conference they preside over. An accepted paper can be assigned to only one conference session inside the same conference. X
* POST: /conference/{conference id}/sessions/{session id}/papers
* A chair can modify any of the four deadlines of a conference. X
* PUT: /conference/{conference id}

c. Author functionalities:

* An author can see all the papers they have personally submitted.x
* GET: /author/papers
* An author can see the status (pending/accepted/rejected) of any paper they have personally submitted.x
* Same as above
* An author can see the paper information of any paper they have personally submitted.x
* Same as above
* An author can upload the camera-ready copy of a paper.
* An author can upload the full paper (PDF or Word).
* An author can submit paper information (title, abstract, topics, keywords, other authors, their emails, their addresses, their phone numbers).x
* POST: /paper/

d. Reviewer functionalities:

* A reviewer can see a list of all papers.x
* GET: /paper/
* A reviewer can see the details (title, abstract, topics, keywords, authors, authors' contact information) of any paper.x
* Same as above
* A reviewer can bid for a paper they are interested in reviewing. X
* POST: /paper/{id}/bids
* A reviewer can specify and modify their topics of interest.x
* PUT: /reviewer/topics
* A reviewer can see their topics of interest.x
* GET: /reviewer/topics
* The system must assign papers automatically to reviewers. X
* PUT: /conference/{id}/papers/assign (as the chair of the conference)
* A reviewer can see a list of all the papers that have been assigned to them to be reviewed. X
* GET: /reviewer/papers
* A reviewer can indicate a conflict of interests found in a paper. X
* POST: /paper/{id}/conflicts
* A reviewer can submit an evaluation to a paper they had to review (accepted/rejected) x
* POST: /paper/{id}/evaluations
* A reviewer can see all the papers they have reviewed. X
* GET: /reviewer/papers/evaluations
* A reviewer can submit a special comment to any paper they have reviewed. Any reviewer can then see the special comment. X
* POST: /paper/{id}/comments

**3.3 Nonfunctional requirements:**

**Usability:**

--

**Reliability:**

The system must be able to restart after system failure.

The system must be able to back-up data and recover data using back-ups.

The system must ensure that username and emails (sign in information) is unique.

The system must ensure that passwords are securely stored.

**Performance:**

The system should be able to display information and execute updates in less than 5 seconds.

The system should be able to generate the documents mentioned in the functional requirements in less than 20 seconds.

**Supportability:**

--

**3.4 Pseudo-requirements:**

**Implementation:**

Use Case Diagrams, Class Diagrams, Activity Diagrams are all constructed using starUML.

The business logic is implemented in Java.

The front-end is written in React.

**Interface:**

--

**Operations:**

As a version control technology, Git is used.

Github is used a host for the remote repository for the versions of the project and for the system specifications.

Github Issues is used for managing and planning system development.

**Packaging:**

--The deployment technologies: to be filled in.

**Legal:**

The user's personal information is strictly confidential. Only the user himself can view it or modify it. User's information will not be leaked to tertiary parties.

**3.5 System model**:

**3.5.1 Scenarios**:

**A. ---------------------------------------------------------------------------------------------------------------------------**

name: uploadPersonalInformation

actor: Jane:User

flow of events:

1. Jane signs in and arrives at the main page

2. On the main page of the website, out of the options listed on a menu, she chooses "Personal Information". A page displaying a table with her personal information appears.

3. She changes the fields 'Last Name' to the value Lee and a few other fields, and saves the changes by interacting with a button. Now the table reloads and displays the updated information.

**B. ---------------------------------------------------------------------------------------------------------------------------**

name: uploadCameraReadyCopy

actor: Jeniffer: Author

flow of events:

1. Jenifer signs in and arrives to the main page.

2. Jenifer selects from the menu the option "See all uploaded papers" and arrives at page with a list of all the papers she has uploaded.

3. She wants to participate in three conferences with a paper she has uploaded only to one conference, "Databases and the Relational Model", as far as she remembers. From the list, she selects her paper, titled "UML to Relational" and inspects the details of the paper. The details of the page contains, besides other information, a list with all the conferences in which her paper has been submitted.

4. In the list of conferences, Jenifer sees that the paper has been submitted to two conferences out of the three she intended to take part in.

5. She selects from the menu the option "Conferences".

6. She selects the last conference she wished to take part in with her paper "UML to Relational".

7. She submits her page by uploading a camera-ready copy, filling in some information related to the paper and accepting terms and conditions.

C: **---------------------------------------------------------------------------------------------------------------------------**

name: modifyDeadlines

actor: Thomas: Chair

context:

The conference managed by Thomas , "Bioinformatics and modern challenges", is scheduled to take place the following year, but Thomas decides that it is high time that he sets the deadlines for the various phases of the conference

flow of events:

1. Thomas signs in and arrives at the main page.

2. . He selects from the menu the option "Conferences". A list of all conferences in the system is displayed, but Thomas is interested only in his own conference.

3. He selects "Bioinformatics and modern challenges" from the list and views the deadlines: "Paper Submission", "Paper Review", "Acceptance Notification", "Uploading Accepted Papers"

4. Thomas modifies the deadlines, and if there are any inconsistencies, they are signaled to Thomas.

D: **---------------------------------------------------------------------------------------------------------------------------**

name: setConferenceTopicOfInterest

actor: Lawrence: Chair

context:

Lawrence decides that, in addition to the already mentioned topics of interest of his conference, "Foo Bar", two new topics are accepted this year, XX and YY, due to a visible surge in the interest for them in the regional community.

flow of events:

1. Lawrence signs in and arrives at the main page.

2. He selects the option "Conferences" from the menu and is presented a list with all the conferences.

3. Lawrence selects "Foo Bar" from the list and inspects the "Topics of interest" of the conference.

4. To the already existent topics, Lawrence adds "XX" and "YY"

**E**: **---------------------------------------------------------------------------------------------------------------------------**

name: decideOnPaperStatus

actor: Sterne: Chair

context:

Sterne is not only an excellent conference manager. He is also a reputed computer scientist, having held himself many conferences over the years. One of his conferences "XYZ" is approaching its 'paper review deadline', yet a paper submitted by another well-known computer scientist, almost as famous as Sterne himself, has not been reviewed yet. Unless Sterne himself accepts the paper, the paper will automatically be evaluated as 'rejected' by the system when the deadline is passed. Sterne skims through the paper with ease and joy, and decides that the paper is a gem and must be accepted into the conference.

flow of events:

1. Sterne signs in and arrives at the main page.

2. He selects the option "Conferences" from the menu and is presented a list with all the conferences.

3. He searches for the conference "XYZ" ,selects the conference from the list, and inspects the list of papers submitted to the conference.

4. Sterne searches for the paper he wants to accept, and registers his evaluation of the paper: "accepted".

5. If he changes his mind, Sterne can "clear" or even explicitly "reject" the paper.

**F. ---------------------------------------------------------------------------------------------------------------------------**

name: assignAcceptedPaperToConferenceSession

actor: Roger: Chair

context:

An accepted paper can be assigned only to one conference session within a conference. Roger wants to assign a batch of accepted papers to specific conference sessions. This option only becomes available for a conference after the "acceptance notification deadline" has passed.

flow of events:

1. Roger signs in and arrives at the main page.

2. Roger selects the option "Conferences" from the menu and is presented a list with all the conferences.

3. From the list, he selects the conference for which he wants to assign accepted papers to conference sessions. Then, Roger inspects the list of sessions for the selected conference.

4. Roger selects a given sessions, to which he can add accepted papers that are unassigned, or from which he unassigned papers, which allows him to assign those papers to other sessions of the conference.

**G. ---------------------------------------------------------------------------------------------------------------------------**

name: submitBasicConferenceInfo

actor: Joseph: Chair

flow of events:

1. Joseph signs in and arrives at the main page.

2. Joseph selects the option "Conferences" from the menu and is presented a list with all the conferences.

3. Joseph filters the list such that he only sees the conferences presided by him.

4. Joseph selects the conference for which he wants to modify the basic information. He inspects the basic conference information associated to the selected conference (if there is any) and modifies it.

**H. ---------------------------------------------------------------------------------------------------------------------------**

name: specifyReviewerTopicsOfInterest

actor: Suzan: Reviewer

context:

Suzan underwent a master's at TUM, and the scope of her interest became wider. She wants to add three new topics, "A", "B", "C" to the topics of interests specified in her account.

flow of events:

1. Suzan signs in and arrives at the main page.

2. Suzan selects the option "Topics of Interest" from the menu. Now, Suzan sees a list with all her specified topic of interest.

3. Suzan adds at the end of the list "A", "B", "C", and then she saves the changes.

**I. ---------------------------------------------------------------------------------------------------------------------------**

name: bidForInterestingPaper

actor: Charlotte: Reviewer

context:

Charlotte has heard from other reviewers about some papers that might interest her. Unless she searches for those papers and bids for them, she might not have the chance to read them.

flow of events:

1. Charlotte signs in and arrives at the main page.

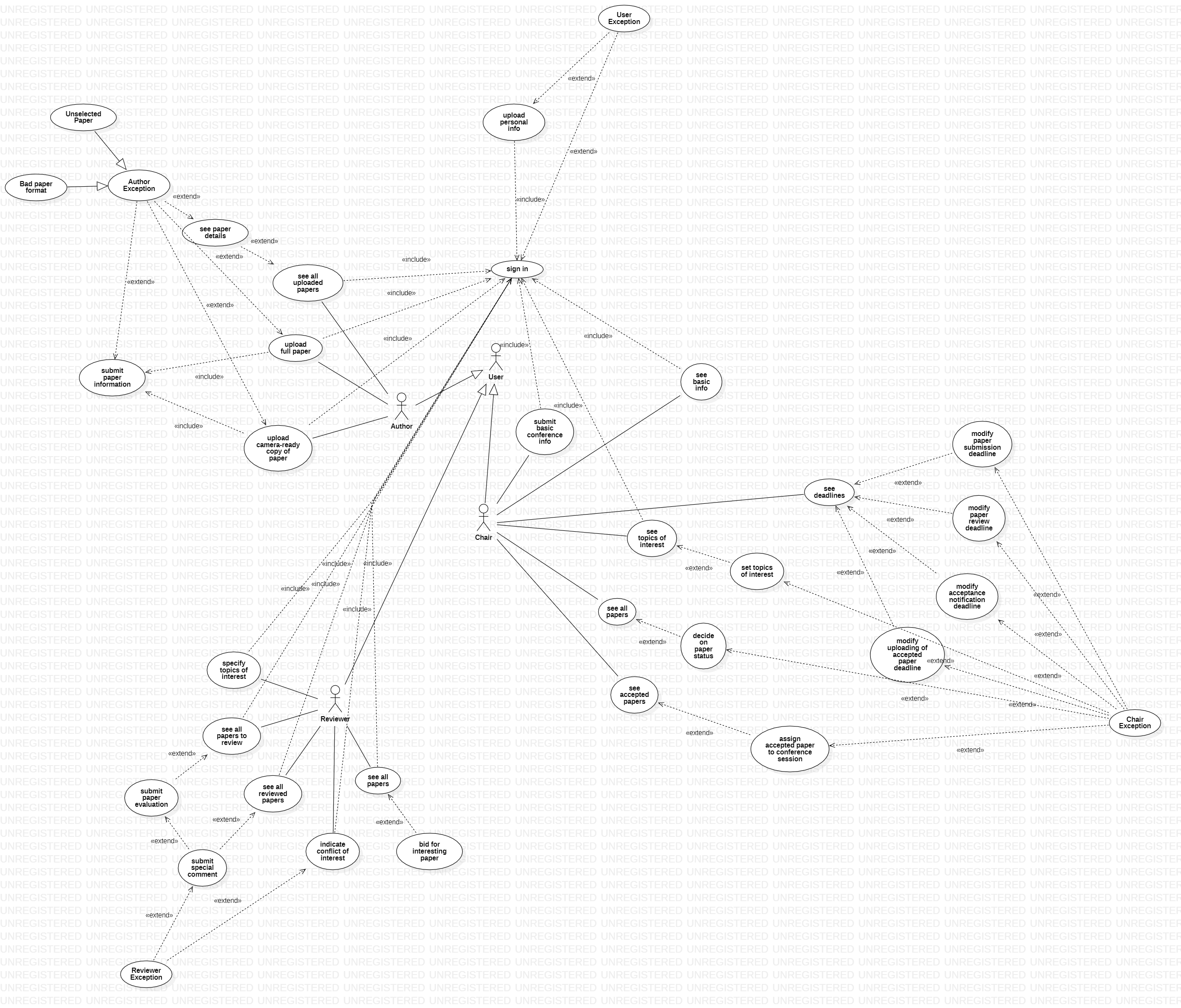
2. Charlotte selects the option "Papers" from the menu, which brings her to a list with all the submitted papers from all the conferences.

3. She scrolls through the list, selecting and inspecting the abstracts of those papers whose titles caught her attention.

4. When she comes across a paper which seems very interesting to her, Suzan bids for the paper with a certain "interest" (for example, values from 1 to 100).

**3.5.2 Functional View:**

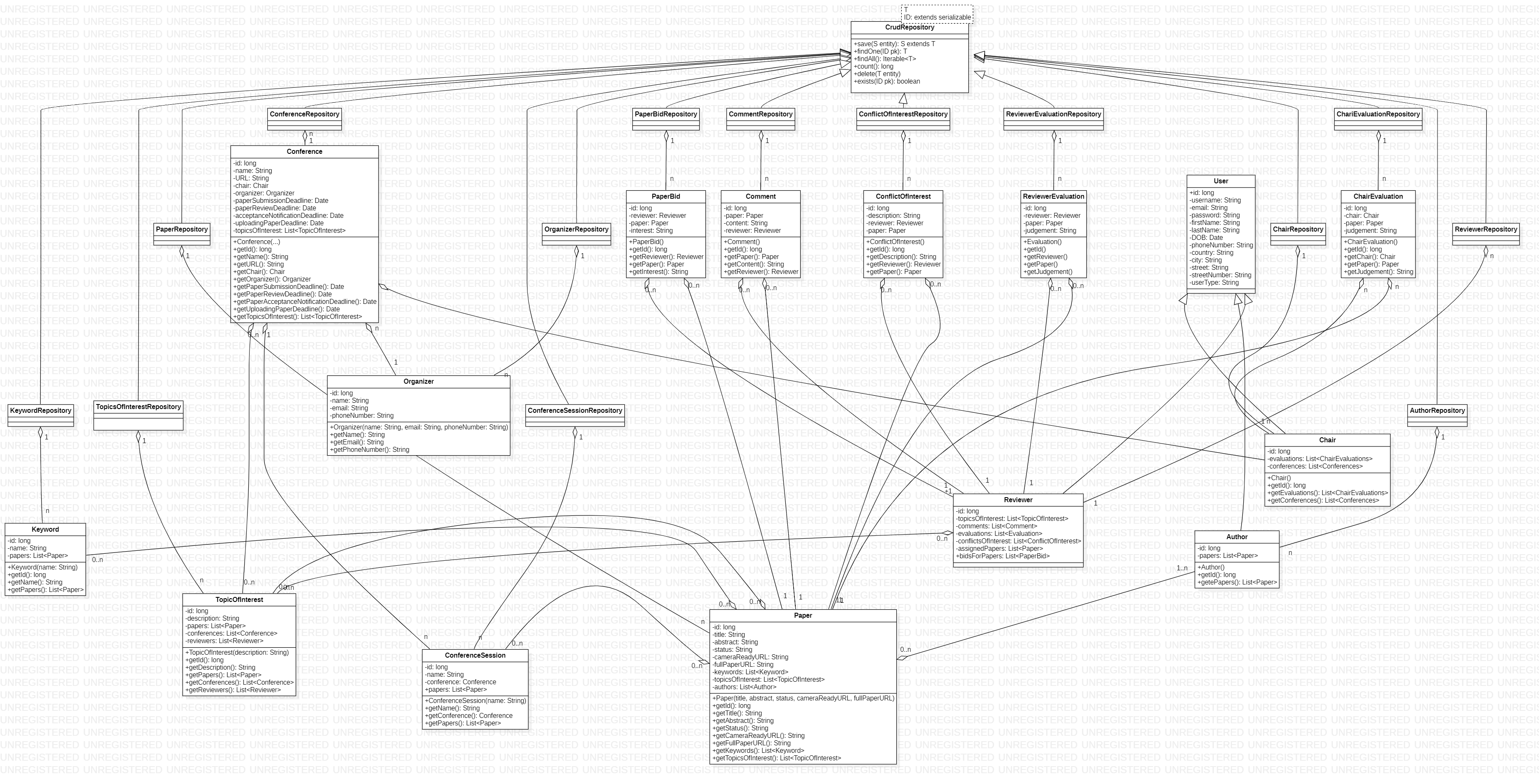
**I. Use Case Diagram**

****

II. Use Case Description -> to be added in the word document, after Tudi and Maria will have finished writing it.

**3.5.3 Structural View:**

**I. Class Diagram**



II. Database Diagrams -> to be added in the word document, after I arrive home.

**3.5.4 Behavioral View:**

**-- To be added**

Activity diagrams. There can be various such diagrams, one for each functionality.

**3.5.5. Interface:**

**-- To be added**

**Squad: Popescu Adrian, Popescu Maria, Popescu Tudor, Savin Radu, Șandor Alexandra – group 926**