



CS 353

Database Management Systems

Project

Project Design Report

Scientific Papers Data Management System

Section 3 / Group 1

Oğuz Kaan Ağac 21503895

Berke Deniz Başaran 21400996

Mert Özerdem 21300835

Ceren Uysal 21501972

Instructor

Cağrı Toraman

REVISED E/R MODEL	3
RELATION SCHEMAS	6
2.1. Organization	6
Relational Model:	6
2.2. Conference	6
2.3. Journal	7
2.4. User	8
2.5. Institution	9
2.6. Publication	9
2.7. Subscriber	10
2.8. Reviewer	10
2.9. Author	11
2.10. Skills	12
2.11. Education	12
2.12. Work Experience	13
2.13. Endorse	14
2.14. Submit	15
2.15. Assign	15
2.16. Review	16
2.17. Subs	17
2.18. Belongs	17
FUNCTIONAL DEPENDENCIES AND NORMALIZATION OF TABLES	19
FUNCTIONAL COMPONENTS	19
4.1. Use Cases / Scenarios	19
4.1.1. User	20
4.1.3 Editor	22
4.1.4 Reviewer	24
4.1.5. Subscriber	25
4.2. Algorithms	26
4.2.1. Subscription Algorithm	26
4.2.2. Endorsement On Skills Algorithm	26
4.2.3. Logical Requirements	26
4.3. Data Structures	26
USER INTERFACE DESIGN AND CORRESPONDING SQL STATEMENTS	27
5.1. Login	27
5.2. Forgot The Password	28
5.3. Register	29
5.4. Search For Standing Conferences	30
5.5. Screen Of Search Result	31

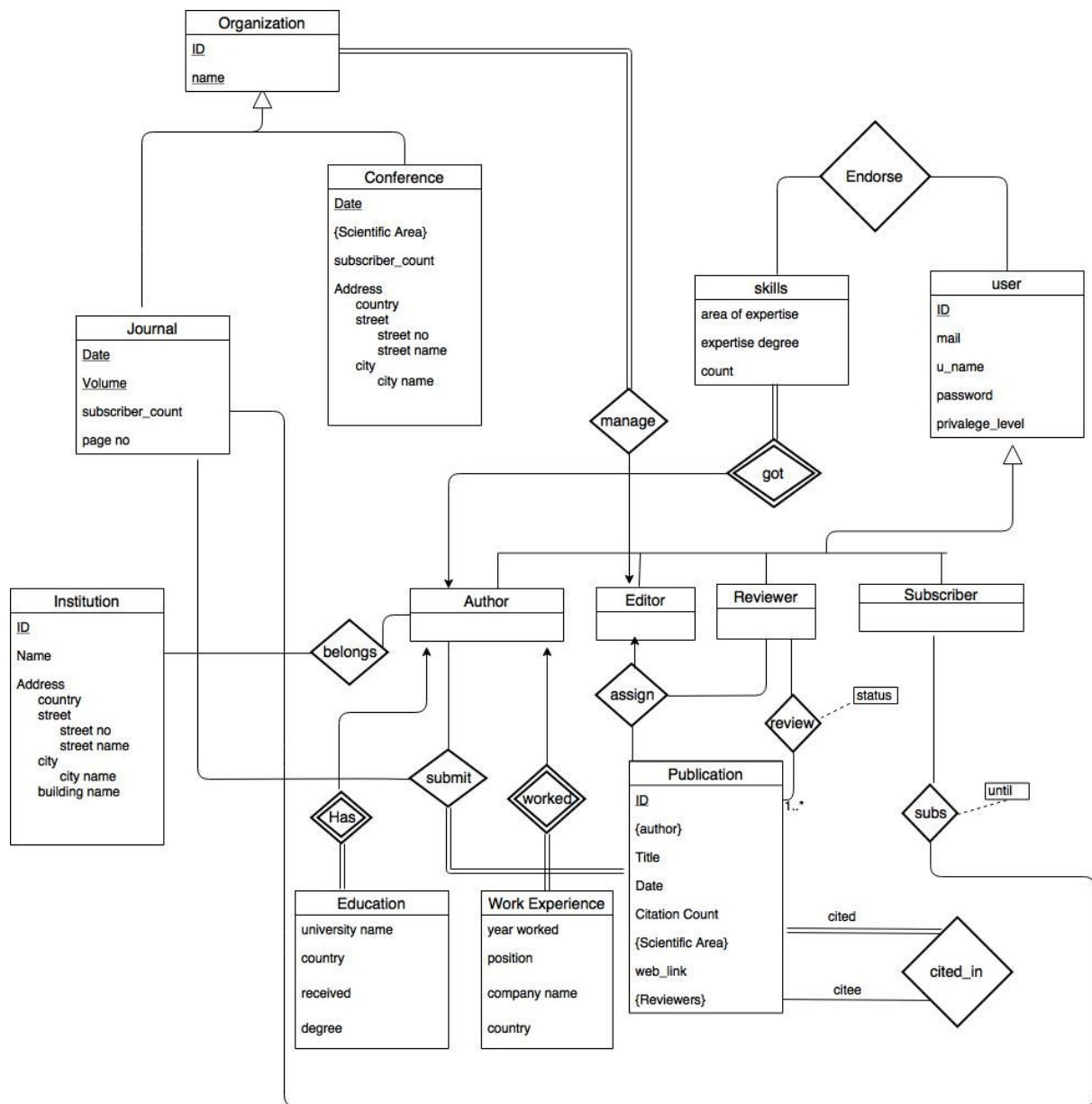
5.6. Advanced Search For Conference	32
5.7. Advanced Search For Journal	33
5.8. Advanced Search For Institution	34
5.9. Advanced Search For Publication	35
5.10. Advanced Search For Author	36
5.11. Profile: Add new research & Edit Info	37
ADVANCED DATABASE COMPONENTS	40
6.1. Views	40
6.1.1 Authors by Institution View	40
6.1.2 Publications by Author Name View	40
6.1.3 Publications by Organizations View	40
6.1.4 Journals by subscription count	41
6.2. Stored Procedures	41
6.3. Reports	44
6.3.1 Number of submitted papers per journal	44
6.4. Triggers	45
6.5. Constraints	46
IMPLEMENTATION PLAN	46
WEBSITE	46

1. REVISED E/R MODEL

According to the feedback from our assistant, there are several changes in the E/R Model in order to make the database design better and more functional. These changes are the following:

- Institution is not a publisher anymore. We have a new relation which is called belongs between Institution and Author in order to indicate whom author work for.
- We have create a new entity called Organization.
- Conference and Journal are not weak entities. We have made is - a relationship between Journal, Conference and Organization.
- Institution is not a weak entity anymore.
- We have renamed the relation which is called e_review as manage.
- We have removed date attribute from manage relation.
- This newly named manage relation is not between Conference, Journal and Editor anymore. It is in between Organization and Editor.
- We have added total participation to Organization in manage relation.
- We removed journal_inst relation between the Journal and Institution.
- We removed hold relation between Institution and Conference.
- In cited_in relation, we have added total participation to cited.
- We have removed conference attribute from subs relation.
- We have removed journal attribute from subs relation.
- We have changed participation type as total participation to Publication in relation called submit.
- We have removed Address entity.
- Adress attribute is added to Conference entity.
- Adress attribute is added to Journal entity.
- Account entity is removed.
- Attributes of Account entity is added as new attrributes to User Entity.
- Submission_list entity is removed.
- Weak relation has which is between, Conference and Submission_list has been removed.
- {Submission} and Submission_Count() attributes of Author is removed.
- Subscribed_journal and subcribed_conference which are attributes of Subscriber has been removed.
- Publication Editor relation which is called assign has been changed as one to many.
- Relation that is called has between Journal and Submission_List is removed.
- {Proficiency} attribute from Author has been removed.
- {Degree} attribute from Author has been removed.
- A new weak entity which is called Work Experience is created.
- A new relation named worked, between Work Experience and Author has been created.
- A new weak entity which is called Education is created.
- A new relation that is called has, between Education and Author has been created.
- We have removed the connection between Publication entity and subs relation.

- A new weak entity which is called Skills has been created.
- A new relation named got, between Skills and Author entities is created.
- We have added new attribute named as until to subs relation.



2. RELATION SCHEMAS

2.1. Organization

Relational Model:

Organization(ID,name)

Functional Dependencies:

None

Candidate Keys:

{{ID, name}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE organizations (  
    ID INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,  
    organizationtitle VARCHAR(64) NOT NULL,  
    creation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

2.2. Conference

Relational Model:

Conference(conference_ID,Name,Conference_Date, Scientific_Area,
subscriber_count, country, street_no, street_name, city_no, city_name)

Functional Dependencies:

conference_ID, Name, Conference_Date, -> Scientific_Area,
subscriber_count, country, street_no, street_name, city_name

Candidate Keys:

{{conference_ID, Name, Conference_Date}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE conferences(  
    conferenceID INT UNSIGNED NOT NULL,  
    conferenceDate DATETIME NOT NULL UNIQUE,  
    subscriber_count INT DEFAULT 0,  
    country VARCHAR(50) NOT NULL,  
    street_no INT,  
    street_name VARCHAR(50),  
    city_name VARCHAR(50) NOT NULL,  
    PRIMARY KEY (conferenceID, conferenceDate),  
    FOREIGN KEY (conferenceID)  
        REFERENCES organizations(ID)  
        ON DELETE CASCADE  
);
```

2.3. Journal

Relational Model:

Journal(journal_ID, title, Journal_Date, volume, subscriber_count, page_no)

Functional Dependencies:

journal_ID, title, Journal_Date, volume -> subscriber_count, page_no

Candidate Keys:

{(journal_ID, title, Journal_Date, volume)}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE journals (  
    ID INT UNSIGNED NOT NULL,  
    volume TINYINT,  
    journalDate DATETIME NOT NULL UNIQUE,  
    subscriber_count INT DEFAULT 0,  
    page_no INT NOT NULL,  
    PRIMARY KEY (journalID, journalDate),  
    FOREIGN KEY(journalID)  
        REFERENCES organizations(ID)  
        ON DELETE CASCADE  
);
```


2.4. User

Relational Model:

User(ID, mail, u_name, password, privilege_level)

Functional Dependencies:

ID -> mail, u_name, password, privilege_level

mail -> ID, u_name, password, privilege_level

u_name -> ID, mail, password, privilege_level

Candidate Keys:

{{ID},{mail},{u_name}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE users (  
    ID INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,  
    mail VARCHAR(64) NOT NULL UNIQUE,  
    username VARCHAR(20) NOT NULL UNIQUE,  
    firstname VARCHAR(50) NOT NULL,  
    middlename VARCHAR(50),  
    lastname VARCHAR(50) NOT NULL,  
    password VARCHAR(20) NOT NULL,  
    privilege_level INT DEFAULT 0,  
    creation_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

2.5. Institution

Relational Model:

Institution(ID, Name, country, street no, street name, city name, building
name)

Functional Dependencies:

ID -> Name, country, street no, street name, city name, building name

Candidate Keys:

{{ID}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE institutions (  
    ID INT UNSIGNED NOT NULL PRIMARY KEY,  
    name VARCHAR(50),  
    country VARCHAR(50) NOT NULL,  
    street_no MIDDLEINT,  
    street_name VARCHAR(50),  
    city_name VARCHAR(50) NOT NULL  
);
```

2.6. Publication

Relational Model:

Publication(ID,author, Title, Date, Citation Count, Scientific Area, web_link,
Reviewers)

Functional Dependencies:

ID -> author, Title, Date, Citation Count, Scientific Area, web_link, Reviewers

Candidate Keys:

{{ID}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE publications (  
    ID INT UNSIGNED NOT NULL AUTO_INCREMENT  
        PRIMARY KEY,  
    publicationtitle VARCHAR(100) NOT NULL,  
    publicationDate DATETIME NOT NULL,
```

```

        citationCount INT DEFAULT 0,
        weblink VARCHAR(100)
    );

```

2.7. Subscriber

Relational Model:

Subscriber(subscriber_ID)

Functional Dependencies:

None

Candidate Keys:

{{subscriber_ID}}

Normal Form:

BCNF

Table Definition:

```

CREATE TABLE subscribed (
    subscriberID INT UNSIGNED NOT NULL,
    organizationID INT UNSIGNED NOT NULL,
    subsUntill DATE,
    FOREIGN KEY (subscriberID)
        REFERENCES users(ID)
        ON DELETE CASCADE,
    FOREIGN KEY (organizationID)
        REFERENCES organizations(ID)
        ON DELETE CASCADE,
    PRIMARY KEY (subscriberID, organizationID)
);

```

2.8. Reviewer

Relational Model:

Reviewer(reviewer_ID)

Functional Dependencies:

None

Candidate Keys:

{{reviewer_ID}}

Normal Form:
BCNF

Table Definition:

```
CREATE TABLE reviews (  
    reviewerID INT UNSIGNED NOT NULL,  
    publicationID INT UNSIGNED NOT NULL,  
    status INT NOT NULL,  
    FOREIGN KEY (reviewerID)  
        REFERENCES users(ID)  
        ON DELETE CASCADE,  
    FOREIGN KEY (publicationID)  
        REFERENCES publications(ID)  
        ON DELETE CASCADE,  
    PRIMARY KEY (reviewerID, publicationID));
```

2.9. Author

Relational Model:
Author(author_ID)

Functional Dependencies:
None

Candidate Keys:
{{author_ID}}

Normal Form:
BCNF

Table Definition:

```
CREATE TABLE authors(  
    publicationID INT UNSIGNED NOT NULL,  
    authorID INT UNSIGNED NOT NULL,  
    FOREIGN KEY (authorID)  
        REFERENCES users(ID)  
        ON DELETE CASCADE,  
    FOREIGN KEY (publicationID)  
        REFERENCES publications(ID)  
        ON DELETE CASCADE,  
    PRIMARY KEY (publicationID, authorID)  
);
```

2.10. Skills

Relational Model:

Skills(ID, area of expertise, expertise degree, count)

Functional Dependencies:

ID -> area of expertise, expertise degree, count

Candidate Keys:

{{ID}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE skills (  
    experties VARCHAR(100) NOT NULL UNIQUE,  
    degree VARCHAR(20),  
    authorID INT UNSIGNED NOT NULL,  
    endorsmentCount INT DEFAULT 0,  
    FOREIGN KEY(authorID)  
        REFERENCES users(ID)  
        ON DELETE CASCADE,  
    PRIMARY KEY (authorID, experties, degree)  
);
```

2.11. Education

Relational Model:

Education(ID, university name, country, received, degree)

Functional Dependencies:

ID -> university name, country, received, degree

Candidate Keys:

{{ID}}

Normal Form:

BCNF

Table Definition:

```

CREATE TABLE educations (
    authorID INT UNSIGNED NOT NULL,
    universityName VARCHAR(100) NOT NULL,
    country VARCHAR(50) NOT NULL,
    year_recieved INT UNSIGNED NOT NULL,
    degree INT NOT NULL,
    FOREIGN KEY (authorID)
        REFERENCES users(ID)
        ON DELETE CASCADE,
    PRIMARY KEY (authorID, universityName, country,
        year_recieved, degree)
);

```

2.12. Work Experience

Relational Model:

Work Experience(author_ID, year worked, position, company name, country)

Functional Dependencies:

author_ID -> year worked, position, company name, country

Candidate Keys:

{{author_ID}}

Normal Form:

BCNF

Table Definition:

```

CREATE TABLE workExperience (
    authorID INT UNSIGNED NOT NULL,
    yearWorked INT UNSIGNED NOT NULL,
    employerName VARCHAR(50) NOT NULL,
    country VARCHAR(50) NOT NULL,
    FOREIGN KEY (authorID)
        REFERENCES users(ID)
        ON DELETE CASCADE,
    PRIMARY KEY (authorID, yearWorked, employerName,
country)
);

```

2.13. Endorse

Relational Model:

Endorse(ID, area of expertise)

Functional Dependencies:

None

Candidate Keys:

{{ID, area of expertise}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE endorsment(  
    endorserID INT UNSIGNED,  
    endorseeID INT UNSIGNED,  
    publicationID INT UNSIGNED,  
    FOREIGN KEY(endorserID)  
        REFERENCES users(ID),  
    FOREIGN KEY(endorseeID)  
        REFERENCES users(ID),  
    FOREIGN KEY(publicationID)  
        REFERENCES publications(ID)  
    ON DELETE CASCADE,  
    PRIMARY KEY (endorserID, endorseeID,  
publicationID)  
);
```

2.14. Submit

Relational Model:

Submit(publication_ID, author_ID, ID, title, journal_date, volume)

Functional Dependencies:

None

Candidate Keys:

{{publication_ID, author_ID, ID, name, date, volume}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE submitted (  
    organizationID INT UNSIGNED NOT NULL,  
    publicationID INT UNSIGNED NOT NULL,  
    FOREIGN KEY (organizationID)  
    REFERENCES organizations(ID)  
    ON DELETE CASCADE,  
    FOREIGN KEY (publicationID)  
    REFERENCES publications(ID)  
    ON DELETE CASCADE,  
    PRIMARY KEY (organizationID, publicationID)  
);
```

2.15. Assign

Relational Model:

Assign(editor_ID, reviewer_ID, publication_ID)

Functional Dependencies:

None

Candidate Keys:

{{editor_ID, reviewer_ID, publication_ID}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE assigns(  

```



```

        editorID INT UNSIGNED NOT NULL,
        reviewerID INT UNSIGNED NOT NULL,
        publicationID INT UNSIGNED NOT NULL,
        FOREIGN KEY (editorID)
            REFERENCES users(ID)
            ON DELETE CASCADE,
        FOREIGN KEY (reviewerID)
            REFERENCES users(ID)
            ON DELETE CASCADE,
        FOREIGN KEY (publicationID)
            REFERENCES publications(ID)
            ON DELETE CASCADE,
        PRIMARY KEY (editorID, reviewerID, publicationID)
    );

```

2.16. Review

Relational Model:

Review(reviewer_ID, publication_ID, status)

Functional Dependencies:

None

Candidate Keys:

{{reviewer_ID, publication_ID, status}}

Normal Form:

BCNF

Table Definition:

```

CREATE TABLE reviews (
    reviewerID INT UNSIGNED NOT NULL,
    publicationID INT UNSIGNED NOT NULL,
    status INT NOT NULL,
    FOREIGN KEY (reviewerID)
        REFERENCES users(ID)
        ON DELETE CASCADE,
    FOREIGN KEY (publicationID)
        REFERENCES publications(ID)
        ON DELETE CASCADE,
    PRIMARY KEY (reviewerID, publicationID)
);

```

2.17. Subs

Relational Model:

Subs(journal_ID, title, Journal_Date, volume, subscriber_ID, until)

Functional Dependencies:

None

Candidate Keys:

{{journal_ID, title, Journal_Date, volume, subscriber_ID, until}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE subscribed (  
    subscriberID INT UNSIGNED NOT NULL,  
    organizationID INT UNSIGNED NOT NULL,  
    subsUntill DATE,  
    FOREIGN KEY (subscriberID)  
        REFERENCES users(ID)  
        ON DELETE CASCADE,  
    FOREIGN KEY (organizationID)  
        REFERENCES organizations(ID)  
        ON DELETE CASCADE,  
    PRIMARY KEY (subscriberID, organizationID)  
);
```

2.18. Belongs

Relational Model:

Belongs(author_ID, institution_ID)

Functional Dependencies:

None

Candidate Keys:

{{author_ID, institution_ID}}

Normal Form:

BCNF

Table Definition:

```
CREATE TABLE belongs (  
    authorID INT UNSIGNED NOT NULL,  
    instituteID INT UNSIGNED NOT NULL,  
    FOREIGN KEY (authorID)  
        REFERENCES users(ID)  
        ON DELETE CASCADE,  
    FOREIGN KEY (instituteID)  
        REFERENCES institutions(ID)  
        ON DELETE CASCADE,  
    PRIMARY KEY (authorID, instituteID)  
);
```

3. FUNCTIONAL DEPENDENCIES AND NORMALIZATION OF TABLES

The Relational Schemas of E/R Model provides normal forms and functional dependencies. There is no normalization or decomposition because all of the relations are in Boyce-Codd Normal Form.

4. FUNCTIONAL COMPONENTS

4.1. Use Cases / Scenarios

Scientific Papers Data Management is responsible for storing, manipulating, and retrieving the information of people who serve as authors in academic environments, and the data of academic papers with certain attributes. Scientific Papers Data Management System has 5 different users. These users are User, Author, Editor, Reviewer and Subscriber. Each user has not only some common services but also user specified ones.

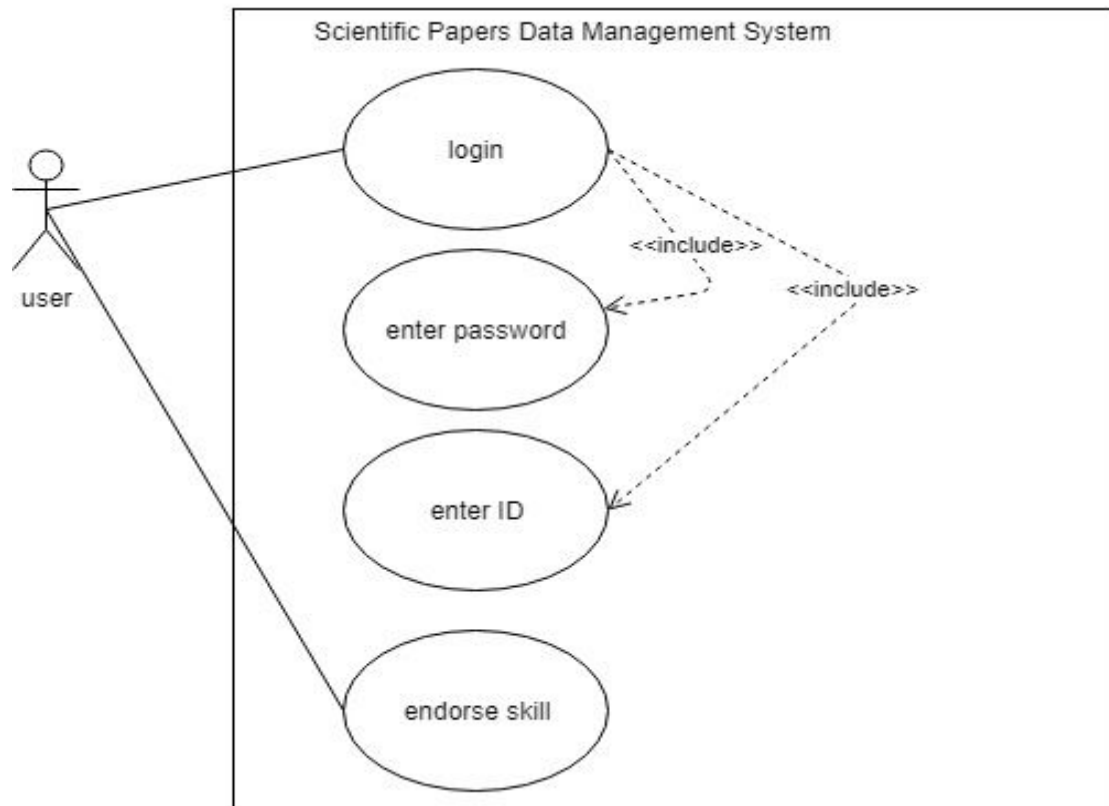
4.1.1. User

Login: Users are able to login to system in order to subscribe an author, submit a publication, review a publication and so forth. To successfully login to the system, users have to enter the correct password and id.

Enter password: Users have to enter their password in order to login to the system successfully.

Enter ID: Users have to enter their ID in order to login to the system successfully.

Approve Skill: Users can see the skills of other users. There is a feature which provides opportunity to all users to endorse skills of other users.



4.1.2 Author

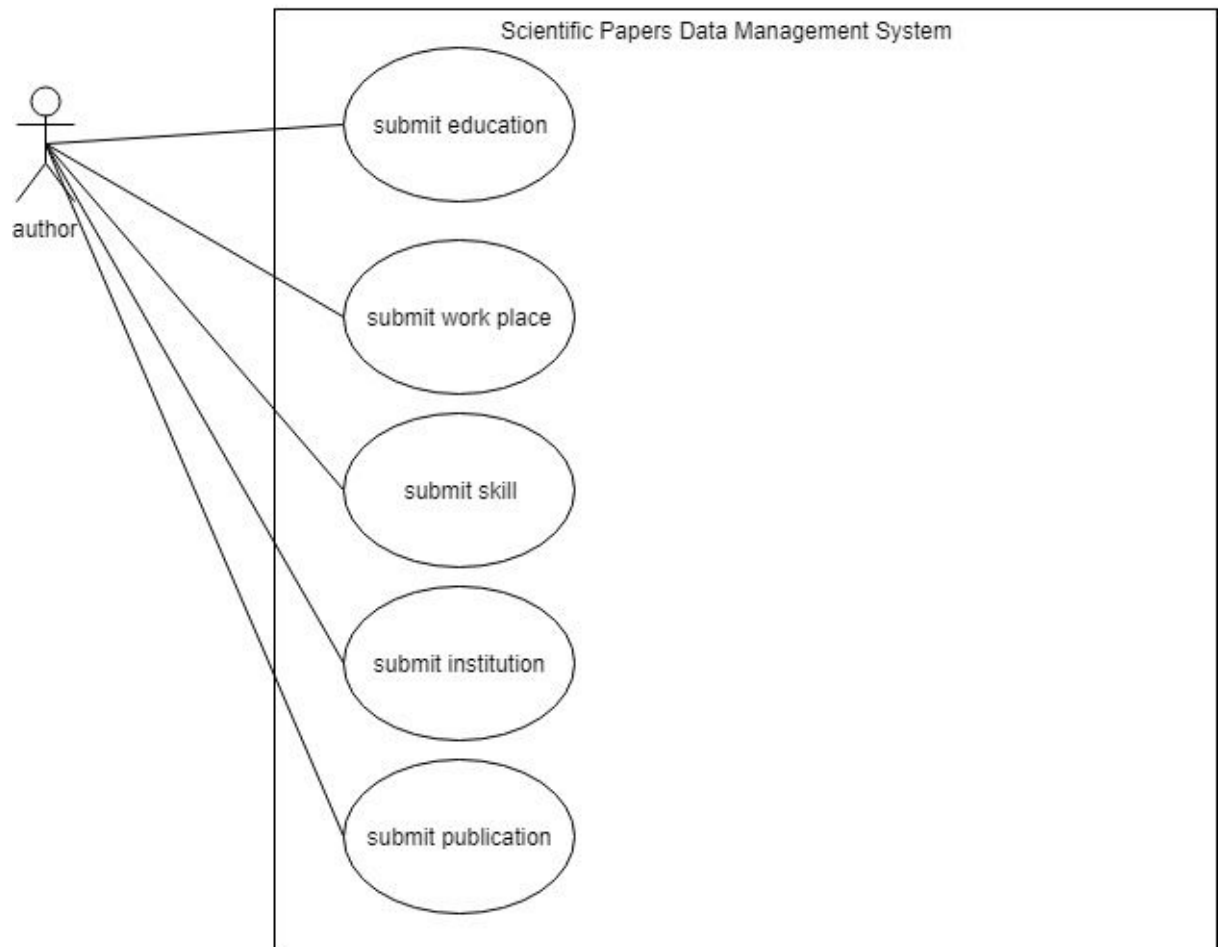
Submit Education: Authors are able to submit their education information.

Submit Work Place: Authors are able to submit their work place.

Submit Skill: Authors can submit their skills.

Submit Institution: Authors can submit their institution.

Submit Publication: Authors can submit any publication of themselves.



4.1.3 Editor

Search Author ID: Editors can search any author id.

View Publication ID: Editors can view any publication of the specified author after searching for the author's id.

View Diploma ID: Editors can view any diploma of the specified author after searching for the author's id.

View Author ID: Editors can view any author id of the specified author after searching for it.

View Institution ID: Editors can view any institution of the specified author after searching for the author's id.

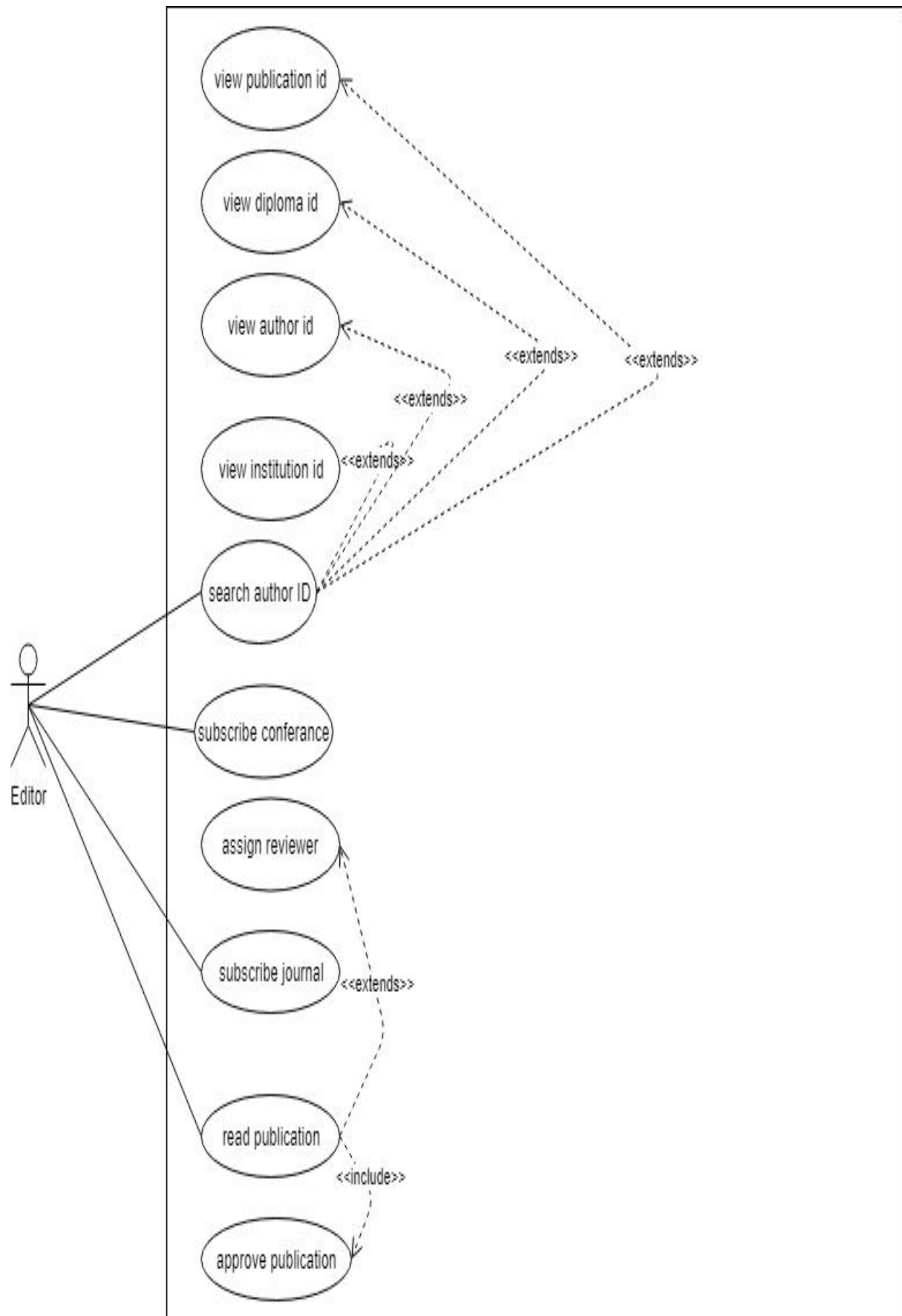
Subscribe Conference: Editors are able to subscribe any conference with respect to their desires.

Assign Reviewer: Editors should assign reviewers to the publications in order to make them ready for publication process after reading the publication.

Subscribe Journal: Editors can subscribe any journal with respect to their pleasure.

Read Publication: Editors are able to read different publications of different authors. With the help of reading publications, the editor decides whether to publish it or not.

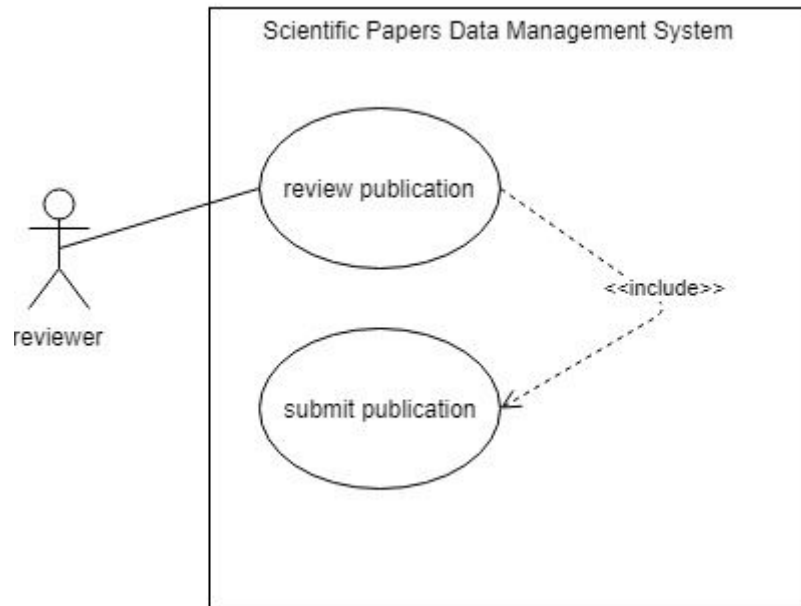
Approve Publication: Editors are able to make decisions about approving a publication or vice versa after they are done with reading the publication.



4.1.4 Reviewer

review publication: Reviewers are responsible for reviewing the publications and give feedbacks according to this process.

submit publication: Reviewers should submit the publications after they are done with reviewing them.

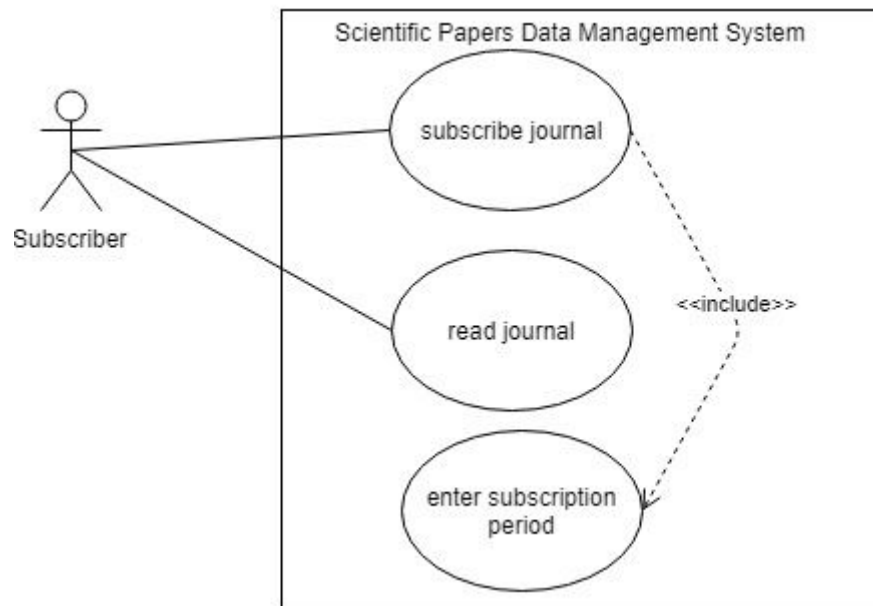


4.1.5. Subscriber

subscribe journal: Subscribers can subscribe any journal with respect to their enjoyment.

read journal: Subscribers are able to read journals.

enter subscription period: Subscribers should enter the subscription period in order to indicate the time duration after subscribing the journal.



4.2. Algorithms

4.2.1. Subscription Algorithm

In our system, subscribers are able to subscribe journals. When a subscriber decides to subscribe a specific journal, he or she should determine a time period in order to indicate the duration of subscription. That value is stored in an attribute named as *until*. If all of the *until*, *subscriber_ID*, *journal_ID*, *title*, *journal_date* and *volume* are compatible with each other, the subscription process is successfully done. Otherwise, process can not be done properly.

4.2.2. Endorsement On Skills Algorithm

In our system, only the authors are able to have skills. This is the reason why only the authors table have skills column. So that In order to have skills, an author must publish a publication. Anyone from the author table who have published before, may have zero or more skills. The skills of the authors must be endorsed by only the users. If a user decides to endorse, both skills table and author table should be updated. Firstly, the user decide which area of expertise of the author to endorse. Then the count of that specific skill that has the same area of expertise, must be incremented by one. Author whose ID is compatible with that updated skills entity, is also affected by this process.

4.2.3. Logical Requirements

Our Scientific Papers Data Management System must not be error prone. There should be information check in order to have valid system, avoid logical errors and so forth. It is needed to check dates, IDs, titles or names whether they are irrelevant or not. Since most of the logical errors are created by them. For example, if an editor assigns a specific publication to a specific reviewer, the reviewer's ID should be equal to the reviewer attribute of Publication after completing the assign process. Also in review process, the ID of the reviewer must be compatible with the ID of the publication in order to have valid data flow.

4.3. Data Structures

Date and Time Type, String Type and Numeric Type are used in the relational schemas which are derived from the E/R Model.

String Types are used to store values which are character compositions. We have used **varchar**. As an example, **varchar** is used to represent mail address of the user.

Date and Time Types are used to store time values. We have used **datetime**. As an example, **datetime** is used to indicate the date of Conference or Journal.

Numeric Types are used to store numeric data. We have used **int**, **tinyint** and **middleint**. As an example, **int** is used to indicate volume from Journal.

5. USER INTERFACE DESIGN AND CORRESPONDING SQL STATEMENTS

5.1. Login

The screenshot shows a web browser window titled "A Web Page" with the address bar displaying "https://www.spdm.com". The main content area of the browser contains the following elements:

- Centered text: "Scientific Papers Data Management System"
- Form fields:
 - "User ID" label above a text input box.
 - "Password" label above a text input box.
- A checkbox labeled "Keep me signed in".
- Two buttons: "Login" and "Reset Password".
- A "Register" button located below the "Login" and "Reset Password" buttons.

Input: @username_or_mail_or_ID, @password

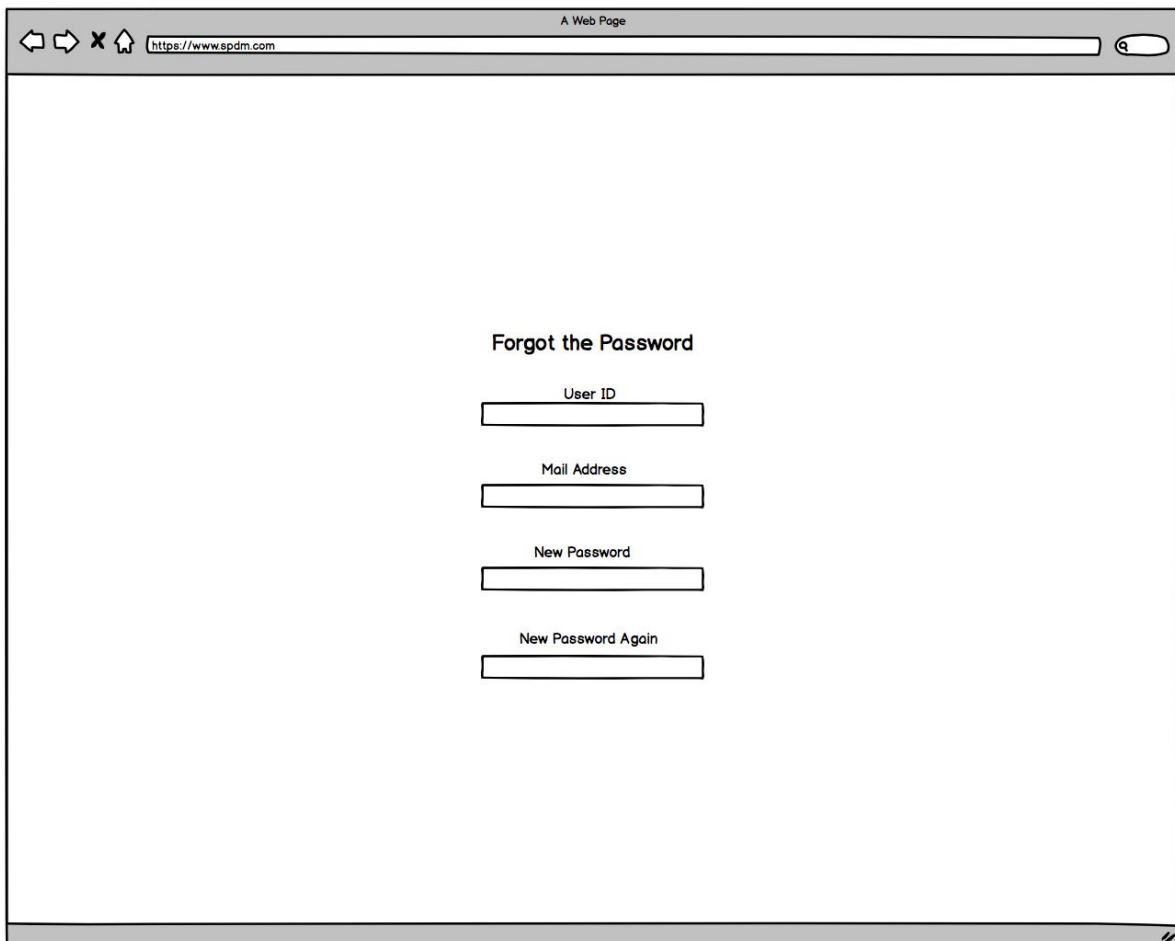
Process: Login page is the first one that application visitor encounters with. In order to use the application, user must login to the system by entering the user ID and password and clicking the login button. Once the boxes are filled, user may choose the "keep me signed in" check box so that credentials are saved by system. Reset password button allows user to assign a new password value if the password is forgotten. From the same page, the visitor may create a new account with the register option provided.

SQL Statements:

Loggin In
SELECT *
FROM users

WHERE (username = @username OR mail = @mail OR ID = @ID)
AND password = @password;

5.2. Forgot The Password



The screenshot shows a web browser window with the address bar displaying 'https://www.spdm.com'. The page title is 'A Web Page'. The main content area displays a form titled 'Forgot the Password'. The form contains four input fields, each with a label above it: 'User ID', 'Mail Address', 'New Password', and 'New Password Again'.

Input: @searchquery, @mail, @ID, @newpassword, @newpasswordagain

Process: User can change the original password from two different pages which navigate user to the forgot the password page. Once the ID, mail address, and new password are entered, forgot the password option is done.

SQL Statements:

Password Retrieval

UPDATE user

SET password = @newpassword

WHERE (mail = @mail OR ID = @ID) AND

@newpassword = @newpassword;

5.3. Register

A Web Page

https://www.spdm.com

Register

User ID

Mail Address

User Name

Password

Password, again

Input: @username, @mail, @ID, @password, @passwordagain

Process: Application is only open for the registered users, so everyone who is willing to use the application must first register by providing user ID, mail address, user name and password.

SQL Statements:

Registration

```
INSERT INTO users(ID, mail, username, password, passwordagain)
VALUES (@ID, @mail, @username, @password, @passwordagain)
WHERE @password = @passwordagain;
```

5.4. Search For Standing Conferences

The screenshot shows a web browser window with the address bar displaying 'https://www.spdm.com'. The page title is 'A Web Page'. The main content area features a navigation bar with three tabs: 'Advanced Search', 'Standing Conferences', and 'Profile'. Below the navigation bar is a search input field with the placeholder text 'Enter the Conference date' and a 'Search' button.

Input: @conferenceDate

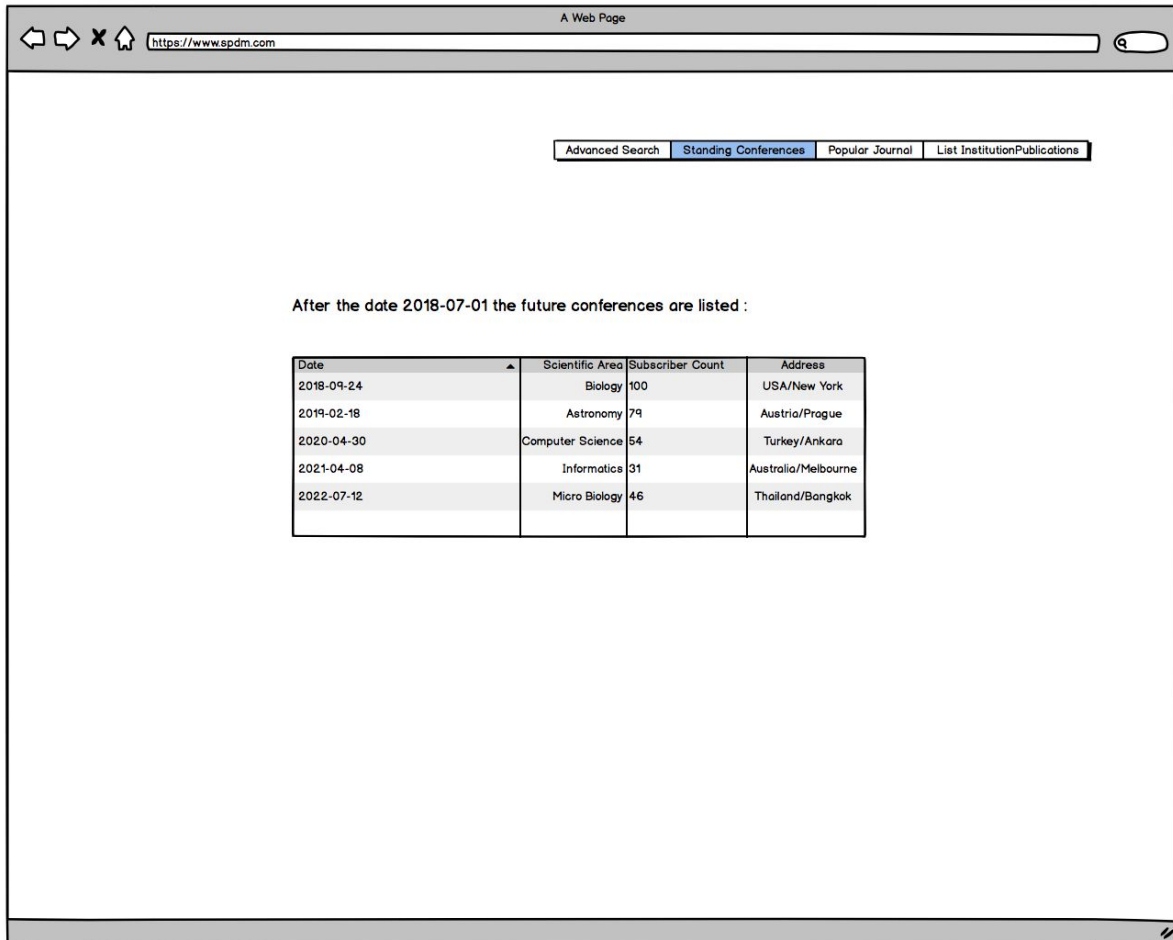
Process: In this page, user can search for the future conferences with a search bar. The results are listed as a bar chart according to the given input.

SQL Statements:

Searching for conferences after a specific date

```
SELECT *  
FROM conferences  
WHERE conferenceDate >= @conferenceDate
```

5.5. Screen Of Search Result



Input: @searchquery

Process: The search results for standing conferences are listed as noticeable.

SQL Statements: In the screen of the search result, there is no need for SQL statements to be executed.

5.6. Advanced Search For Conference

Advanced Search for Conference

Date

Name

Location

Scientific Area

Input: @conferenceID, @name, @country, @scientific_area

Process: In the advanced search page, there are several tabs listed as conference, journal, institution, user, publication, author. Alongside the advanced search option, user may search for other type of elements in system with a keyword to be provided and by clicking the corresponding search button. In this specific case conference is chosen for the advanced search. To search conference, date, name, location, and scientific area must be specified.

SQL Statements:

Searching for conferences

```
SELECT *  
FROM conferences  
WHERE conferenceID = @conferenceID AND name = @name AND  
country = @country AND scientific_area = @scientific_area;
```

5.7. Advanced Search For Journal

The screenshot shows a web browser window with the URL <https://www.spdm.com>. The page has a navigation bar with three tabs: "Advanced Search", "Standing Conferences", and "Profile". The "Advanced Search" tab is selected. Below the navigation bar, the page is titled "Advanced Search for Journal". There are three input fields for "Title", "Date", and "Volume", each followed by a "Search" button. Below these fields, there are five more input fields, each with a magnifying glass icon and a label: "Conference : keyword", "Institution : keyword", "User : keyword", "Publication : keyword", and "Author : keyword". At the bottom, there is a section titled "Select the item for advanced search" with a row of tabs: "Conference", "Journal", "Institution", "User", "Publication", and "Author". The "Journal" tab is selected.

Input: @title, @volume, @journalDate

Process: For the advanced search of journal, title, date, and volume must be specified in the input bars listed at top.

SQL Statements:

Searching for Journals

```
SELECT *  
FROM journals  
WHERE title = @title AND volume = @volume AND  
      journalDate = @journalDate;
```

5.8. Advanced Search For Institution

The screenshot shows a web browser window with the address bar displaying 'http://www.spdm.com'. The page title is 'A Web Page'. The main content area has a navigation bar with four tabs: 'Advanced Search' (selected), 'Standing Conferences', 'Popular Journal', and 'List InstitutionPublications'. Below the navigation bar, the heading 'Advanced Search for Institution' is displayed. The search form includes three input fields for 'ID', 'Name', and 'Type', each followed by a 'Search' button. Below these fields, there are five rows of search criteria, each with a dropdown menu and a 'Search' button: 'Conference : keyword', 'Journal : keyword', 'User : keyword', 'Publication : keyword', and 'Author : keyword'. At the bottom, there is a section titled 'Select the item for advanced search' with a row of tabs: 'Conference', 'Journal', 'Institution' (selected), 'User', 'Publication', and 'Author'.

Input: @ID, @name, @type

Process: For the advanced search of ID, name, and type must be specified in the input bars listed at top.

SQL Statements:

Searching For Institution

```
SELECT *  
FROM institutions  
WHERE (ID = @ID OR name = @name ) AND type = @type;
```

5.9. Advanced Search For Publication

The screenshot shows a web browser window with the URL <https://www.spdm.com>. The page has a navigation bar with three tabs: "Advanced Search", "Standing Conferences", and "Profile". The "Advanced Search" tab is active. Below the navigation bar, the title "Advanced Search for Publication" is displayed. The main content area contains several input fields for search criteria: ID, Author, Web link, Scientific Area, Title, and Date. Each field is followed by a "Search" button. Below these fields, there are five more search options, each with a dropdown menu and a "Search" button: "Conference: keyword", "Journal: keyword", "Institution: keyword", "User: keyword", and "Author: keyword". At the bottom, there is a section titled "Select the item for advanced search" with a horizontal menu containing the following items: "Conference", "Journal", "Institution", "User", "Publication", and "Author". The "Publication" item is currently selected.

Input: @publication_ID, @author, @title, @publicationDate, @scientific_area, @weblink

Process: For the advanced search of ID, author, web link, scientific area, title, and date must be specified in the input bars listed at top.

SQL Statements:

Searching for specific publication

```
SELECT *
FROM publications
WHERE publication_ID = @publication_ID AND author = @author AND
      title = @title AND publicationDate = @publicationDate AND
      scientific_area = @scientific_area AND weblink = @weblink;
```

5.10. Advanced Search For Author

Input: @authorID, @degree, @position, @company_name, @country

Process: For the advanced search of ID, degree, position, company name, and country must be specified in the input bars listed at top.

SQL Statements:

Searching for specific author

```
SELECT *
FROM authors
WHERE authorID = @authorID AND degree = @degree AND
      position = @position AND company_name = @company_name AND
      country = @country;
```

5.11. Profile: Add new research & Edit Info

Advanced Search Standing Conferences Profile

Profile

ID : CagriToraman

Degree : Dr.

Position : Instructor

Company Name : Bilkent University

Country : Turkey

Mail: ctoraman@cs.bilkent.edu.tr

Area of Expertise :

Data Science
Information Retrieval
Machine Learning
Social Network
Text Mining

Edit Profile Information Set New Password

Title

Date

Web Link

Scientific Area

Other Authors

+
Add new research

Input:

- **For “Edit Profile Info”:**
@newID, @newdegree, @newposition, @newcompany_name,
@newcountry, @newareaOfExpertise, @newmail
- **For “Add new research”:**
@Title, @ID, @author, @date, @web_link

SQL Statements:

- Edit Profile Info
- Add new research

Process: Users may edit the related information in any time. They can edit their ID, degree, position, company name, country, and mail. There are two additional functionalities of the profile page. One is that user can add a new

research once the title, date, web link, scientific area, and other authors, who contributed to the paper, are specified at the given box. The other functionality allows user to reset profile's password.

SQL Statements:

Edit Profile Info:

Change user ID

```
UPDATE user  
SET ID = @newID  
WHERE ID = @ID
```

Change company name

```
UPDATE user  
SET ID = @newcompany_name  
WHERE ID = @ID
```

Change position

```
UPDATE user  
SET ID = @newposition  
WHERE ID = @ID
```

Change degree

```
UPDATE user  
SET ID = @newdegree  
WHERE ID = @ID
```

Change mail

```
UPDATE user  
SET ID = @newmail  
WHERE ID = @ID
```

Change area of expertise

```
UPDATE area of expertise  
SET ID = @newareaOfExpertise  
WHERE ID = @ID
```

Add new research:

```
INSERT INTO publications  
VALUES(@Title, @ID, @author, @date, @web_link, @author)  
WHERE ID = @ID
```


6. ADVANCED DATABASE COMPONENTS

6.1. Views

6.1.1 Authors by Institution View

List all authors by their institution.

```
CREATE VIEW authors_by_institution AS
SELECT I.name, U.lastname, U.firstname, IFNULL(U.middlename, "")
FROM belongs B LEFT JOIN users U
      ON B.authorID = U.ID
     LEFT JOIN institutions I
      ON B.instituteID = I.ID
ORDER BY I.name;
```

6.1.2 Publications by Author Name View

List all publications by their authors

```
CREATE VIEW journals_by_authurname AS
SELECT U.lastname, U.firstname, IFNULL(U.middlename, ""),
       P.publicationtitle, P.publicationDate, O.organizationtitle
FROM authors A LEFT JOIN submitted S
      ON A.publicationID = S.publicationID
     LEFT JOIN organizations O
      ON O.ID = S.organizationID
     LEFT JOIN publications P
      ON O.ID = P.ID
     LEFT JOIN users U
      ON A.authorID = U.ID
ORDER BY U.lastname, U.firstname, IFNULL(U.middlename, "");
```

6.1.3 Publications by Organizations View

List all publications by their submitted organizations.

```
CREATE VIEW publications_by_organizations AS
SELECT P.publicationtitle, O.organizationtitle
FROM submitted S LEFT JOIN publications P
      ON P.ID = S.publicationID
     LEFT JOIN organizations O
      ON S.organizationID = O.ID
ORDER BY O.organizationtitle;
```

6.1.4 Journals by subscription count

List all journals by their popularity.

```
CREATE VIEW journals_by_subscriptionCount AS
SELECT O.title, J.journalDate, COUNT(*) as subscount
FROM subscribed S LEFT JOIN journals J
      ON S.organizationID = J.journalID
LEFT JOIN organizations O
      ON J.journalID = O.ID
GROUP BY J.journalID, J.journalDate;
```

6.1.5 Organizations by managing editor's name view

List all organizations by their editor's name

```
CREATE VIEW org_by_editors AS
SELECT U.lastname, U.firstname, IFNULL(U.middlename, ""), O.organizationtitle
FROM manages M LEFT JOIN users U
      ON M.editorID = U.ID
LEFT JOIN organizations O
      ON O.ID = M.organizationID
ORDER BY U.lastname, U.firstname, IFNULL(U.middlename, "");
```

6.2. Stored Procedures

Procedure for getting every information about a user

@INPUT: USER ID

@OUTPUT: ID, MAIL, USERNAME, PASSWORD, PRIVILIDGE, CREATION DATE

```
DELIMITER //
CREATE PROCEDURE getUser(
    IN userID INT UNSIGNED
)
BEGIN
    SELECT * FROM users WHERE ID = userID;
END //
DELIMITER ;
```

Procedure for inserting new subscription information

@INPUT: USER ID, PULICATION ID, DATE TO END OF SUBSCRIPTION

@OUTPUT NONE

```

DELIMITER //
CREATE PROCEDURE subscribe (
    IN userID INT UNSIGNED
    IN subpublicationID INT UNSIGNED
    IN untill DATE
)
BEGIN
    INSERT INTO subscribes(subscriberID, publicationID,
subsUntill)
        VALUES      (userID, subpublicationID, untill);
    END //
DELIMITER;

```

Procedure for inserting new assignment information
@INPUT EDITOR ID, REVIEWER ID, PULICATION ID
@OUTPUT NONE

```

DELIMITER //
CREATE PROCEDURE assign (
    IN inputEditorID INT UNSIGNED
    IN inputReviewerID INT UNSIGNED
    IN inputPublicationID INT UNSIGNED
)
BEGIN
    INSERT INTO assigned(editorID, reviewerID, publicationID)
        VALUES (inputEditorID, inputReviewerID, inputPublicationID);
    END //
DELIMITER;

```

Procedure for inserting new citation information
@INPUT ID OF THE CITING PAPER, ID OF THE CITED PAPER
@OUTPUT NONE

```

DELIMITER //
CREATE PROCEDURE cite (
    IN citedID INT UNSIGNED
    IN citeeID INT UNSIGNED
)
BEGIN
    INSERT INTO cited(citedPublicationID, citingPublicationID)
        VALUES (citedID, citeeID);
    END //
DELIMITER;

```

Procedure for inserting new submission information

@INPUT ORGANIZATION ID, SUBMISSION ID

@OUTPUT NONE

DELIMITER //

```
CREATE PROCEDURE submit(  
    IN inputOrganizationID  
    IN inputPublicationID  
)  
    BEGIN  
        INSERT INTO submitted(organizationID, publicationID)  
        VALUES (inputOrganizationID, inputPublicationID);  
    END //  
DELIMITER ;
```

Procedure for inserting new management information

@INPUT ORGANIZATION ID, EDITOR ID

@OUTPUT NONE

DELIMITER //

```
CREATE PROCEDURE manage (  
    IN inputOrganizationID  
    IN inputEditorID  
)  
  
    BEGIN  
        INSERT INTO manages(organizationID, editorID)  
        VALUES (inputOrganizationID, inputEditorID);  
    END\\  
DELIMITER;
```

6.3. Reports

6.3.1 Number of submitted papers per journal

```
SELECT O.organizationtitle, J.journalDate, COUNT(*)
FROM journals J LEFT JOIN organizations O
      ON J.journalID = O.ID
LEFT JOIN submitted S
      ON J.journalID = S.organizationID
GROUP BY O.organizationtitle, J.journalDate;
```

6.3.2 Number of authors per institute

```
SELECT I.name, COUNT(*)
FROM belongs B LEFT JOIN instutitons I
      ON B.instituteID = I.ID
GROUP BY I.ID;
```

6.3.3 Number of published papers per author

```
SELECT U.lastname, U.firstname, IFNULL(U.middlename, ""), COUNT(*)
FROM submitted S LEFT JOIN authors A
      ON S.publicationID = A.publicationID
LEFT JOIN users U
      ON A.authorID = U.ID
GROUP BY A.authorID;
```

6.3.4 Average number of published papers per journal

```
SELECT O.organizationtitle, J.journalDate, AVG(*)
FROM submitted S LEFT JOIN journals J
      ON S.organizationID = J.journalID
LEFT JOIN organization O
      ON J.journalID = O.organizationID
GROUP BY O.organizationtitle, J.journalDate;
```

6.3.5 Number of publications per scientific area

```
SELECT P.scientificarea, COUNT(*)
FROM submitted S LEFT JOIN publications P
      ON S.publicationID = P.ID
GROUP BY P.scientificarea;
```

6.3.6 The most popular conference

```
SELECT O.organizationTitle, C.subscriber_count
FROM conferences C LEFT JOIN organizations O
      ON C.conferenceID = O.ID
ORDER BY C.subscriber_count DESC LIMIT 1;
```

6.3.7 The most popular journal

```
SELECT O.organizationTitle, J.subscriber_count
FROM journals J LEFT JOIN organizations O
      ON j.journalID = O.ID
ORDER BY C.subscriber_count DESC LIMIT 1;
```

6.3.8 Number of reviewers per publication

```
SELECT RW.publication, COUNT(*)
FROM reviewer RW LEFT JOIN review RE
      ON RE.ID = RW.ID
GROUP BY RE.publication;
```

6.3.9 The most popular author

```
SELECT A.ID
FROM author A LEFT JOIN submitted S
      ON S.ID = A.ID
ORDER BY S.citation_count DESC LIMIT 1;
```

6.3.10 The most privileged user

```
SELECT U.privilage_level
FROM users U
ORDER BY U.privilage_level DESC LIMIT 1;
```

6.3.11 Number of subscribers per conference

```
SELECT C.subscriber_count COUNT(*)
FROM conferences C NATURAL JOIN organizations O
GROUP BY C.subscriber_count;
```

6.4. Triggers

- When a publication is accepted by any of the conferences or journals, it will be removed from the submission list so that it can be submitted to multiple publications but can only be published once.
- When a user subscribes a journal or a conference, subscription count will be incremented

- When a user unsubscribes a journal or a conference, subscription count will be decremented
- When another author endorses another author's expertise, the endorsement count will be incremented
- Submissions can only be made to the future publications.

6.5. Constraints

- Every conference and journal must have at least one editor.
- For the papers, authors and date cannot be null.
- Conferences must have valid address.
- Skills of the authors must be endorsed by other users.
- Institutions must have valid addresses.
- User emails must be unique and valid.

7. IMPLEMENTATION PLAN

In our project, the data flow is managed and controlled by using MySQL Server. In order to implement the web service, we are planning to use HTML, Javascript, Django Python, Bootstrap. For the back-end, we will use PHP.

8. WEBSITE

The link of GitHub page of the project is the following:

<https://github.com/cerenuysal/CS-353>