CS 353 – Database Systems

Scientific Papers Data Management System

Project Final Report

Group 3

Project URL:

dijkstra.ug.bcc.bilkent.edu.tr/~kaan.sancak/scientific-papers-database

Ali Sabbagh	**21500269**
Kaan Sancak	**21502708**
Kanan Asadov	**21503382**
Sabit Gökberk Karaca	**21401862**

Table of Contents

1. Description

Scientific Papers Data Management is system is a web-based application for keeping track of scientific publications, conferences, journal and institutions. There are 4 different types of users in the system: regular subscribers, reviewers, authors, editors. All users can search, download, see details of publications and subscribe for a particular journal. Additionally authors can make new submissions to publish their latest work. If their submission gets approved by reviewers and editors, they can publish their work. Reviewers are able to receive invitations to review submission from editors. They can reject an invitation or accept it by writing a review. Editors are able to send review request to reviewers and they can reject or approve a submission according to the feedback. Editors and authors can have multiple expertise fields. They can only make/review submissions related to their field of expertises. There are two types of publishers in the application: conferences and journals. Audiences can attend conferences that are not passed. On the other hand, journals publishes their volumes if their editors approve journal end.

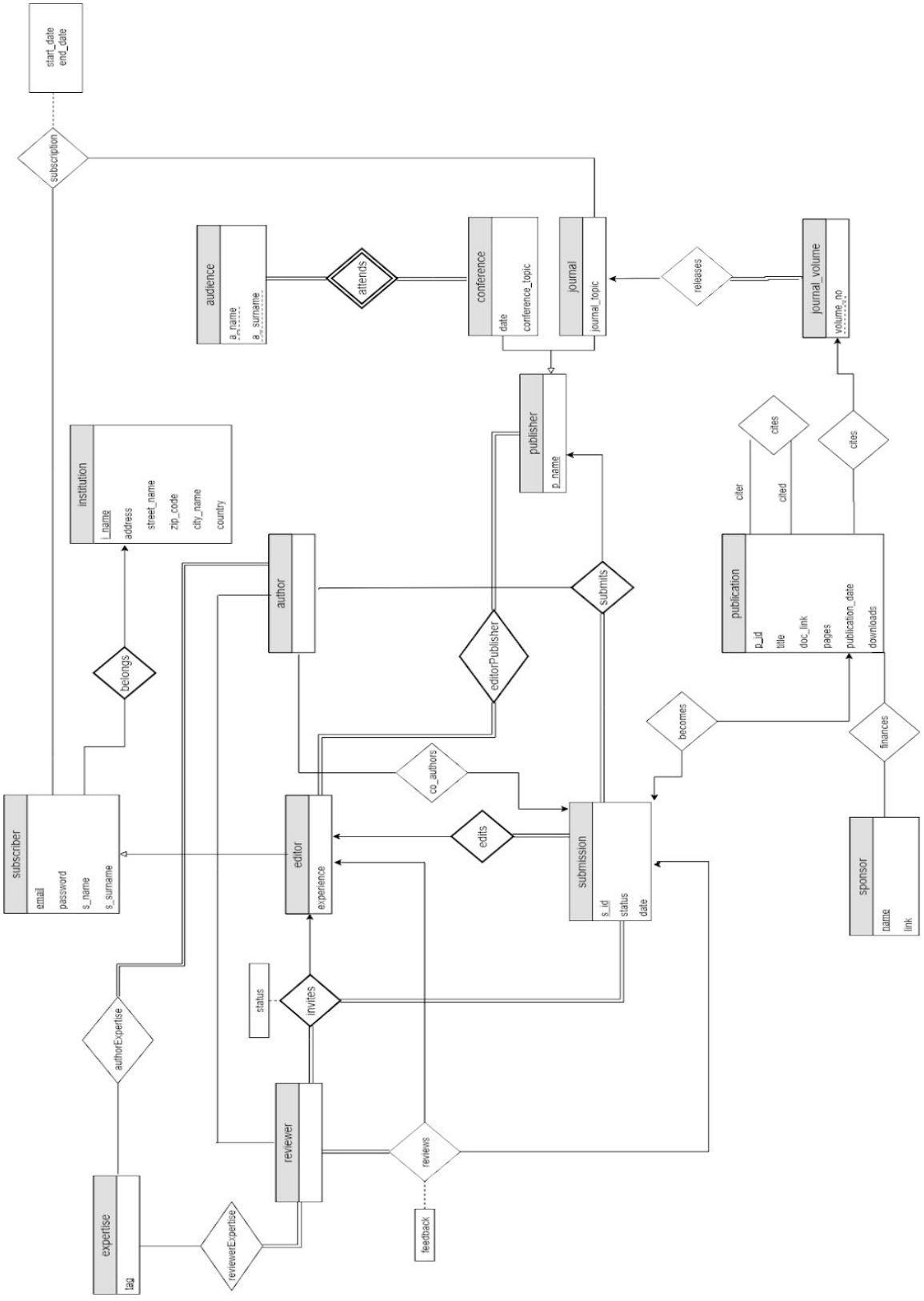
Ali Sabbagh contributed to the database analysis and design phases and ER diagram representation. For the implementation, he worked on the tasks related to, signup page, conference registration, editor-submission, and author-submission pages. He took care of these pages front-end, back-end, and database connection to them.

Kaan Sancak has created the database design and implemented a java program to create tables and add data. He wrote 4 stored procedures and 1 trigger for advanced sql. With his team-mates he contributed in design and implementation different back-end files, author, conference, find-conference, journal, publisher, main user page and index pages.

Kanan Asadov contributed to the ER diagram representation and took care of the user interface during the design stage. During the implementation, he worked on signin, reviewer-submission, editor-submission pages finishing their front/back-ends. He also worked on journal and submission pages adding extra features including their sql queries etc.

Sabit Gökberk Karaca worked on the database design, entities and their relationships. He has worked on a java application to automate the database initialization process. After completing the initial works, he has worked on the graphical user interface, sql queries and database connection of author, conference, audience, publisher, journal, main, index and publication pages.

2. Final E/R Diagram



3. Final Schemas

audience(p_name, a_name, a_surname)

FK: p_name references conference(p_name)

author(email)

FK: email references subscriber(email)

authorExpertise(email, tag)

FK: email references subsriber(email)
FK: email references expertise(tag)

cites(citer, cited)

FK: citer references publication(p_id)
FK: cited references publication(p_id)

co_authors(s_id, email)

FK: s_id references submission(s_id)
FK: email references author(email)

conference(date, conference_topic, p_name)

FK: p_name references publisher(p_name)

editor(experience, email)

FK: email references subscriber(email)

editorPublisher(email, p_name)

FK: email references editor(email)
FK: p_name references p_name

expertise(tag)

finances(name, p_id)

FK: name references sponsor(name)
FK: p_id references publication(p_id)

institution(i_name, stree_name, zip_code, city_name, country)

invites(reviewer_email, editor_email, s_id, status)

FK: editor_email references editor(email)
FK: reviewer_email references reviewer(email
FK: s_id references submission_id

journal(journal_topic, p_name)

FK: p_name references publisher(p_name)

journal_volume(p_name, volume_no)

FK: p_name references publisher(p_name)

publication(p_id, title, s_id, pages, publication_date, doc_link, downloads)

FK: s_id references submission(s_id)

published_in(p_name, volume_no, p_id)

FK: p_name references journal_volume(p_name)
FK: volume_no reference journal_volume(volume_no)
FK: p_id references publication(p_id)

publisher(p_name)

reviewer(email)

FK: email references subscriber(email)

reviewerExpertise(email, tag)

FK: email references reviewer(email)
FK: tag references expertise(tag)

reviews(reviewer_email, editor_email, s_id)

FK: reviewer_email references reviewer(email)
FK: editor_email references editor(email)
FK: s_id references submission(s_id)

sponsor(name, link)

submission(s_id, email, status, title, doc_link, date)

FK: email references editor(email)

submits(email, s_id, p_name)

FK: email references author(email)
FK: s_id references submission(s_id)
FK: p_name references publisher(p_name)

subscriber(email, i_name)

FK: i_name references institution(i_name)

subscription(email, p_name, start_date, end_date)

FK: p_name references journal(p_name)
FK: email references subscriber(email)

4. Implementation Details

When the team was deciding the tools and technologie to use when developing the project, we all agreed that it's better to use a language/tool that is new to us, so this project can serve as a chance to learn it deeply. Hence, we used php for the backend. For the database design and maintenance we used MySQL. We wrote a java program that we used to drop and recreate the table with dummy data whenever needed.

For the front-end, HTML and CSS3 were used to build pages structures. Bootstrap 4 was used for styling. Javascript was used to support the front end, and to make user experience more enjoyable. jQuery library was used for easier traversing of the HTML DOM, and handling events. The database, and the web application pages are all hosted on one of our team members account on bilkent dijkstra server.

For some pages like signup, author-submission, editor-submission, and conference pages, we found the problem that depending on the user interaction, our pages would have some different functionalities, or load different HTML accordingly. For that, and to have a better user experience, with less data being transferred to and from server, we decided to use Ajax with javascript to make asynchronous calls to our server and load/send data dynamically to the needed page. This was immensely useful to solve the popup's pages problem, where on a button click a popup appears which has data fetched from database. Which would be hard to implement if only calling php normally.

As the project went on, more pages were implemented and dependencies between them started to occur, and the need from one team member to have access to another's assigned pages was needed. Then we decided to use git for version controlling. All our pages were pushed to a repository, then modifying the codes, fixing bugs, and linking the pages together (fixing hyperlinks, and setting session variables) was much easier and possible to do by any team member.

One problem we faced was the synchronization of data among team; to solve this problem we set up our database on one of team member's dijkstra account. So, we could all work synchronously. Another problem was the design the mistakes at E/R, everytime we faced this issue we had to re-think about design again and made changes in the implementation which took a lot of our time. Lastly, working with HTML, PHP and JavaScript was challenging for us, since we did not have that much experience in these languages, we had to do a lot of practises to learn them.

5. Advanced Database Features

In this project we have used Triggers, Stored Procedures, Constraints and Views. Some examples of these advanced database features are provided in this section.

- **Trigger**
  - If the submission has been reviewed by all reviewers, submission status has been changed.
  - If the reviewer accepts the review invitation and gives a feedback, the entry for this invitation is removed from invitations table.
  - If a submission is approved to be published, a new entry is inserted into publication table.
- **Stored Procedures**
  - **find_author_total_citations(email, @author_total_cited)**

This procedure is used in order to calculate the total number of citations that are made to publications of the specified author.

```
set @author_total_cited = 0;
call gokberk_karaca.find_author_total_citations('email1', @author_total_cited);
select @author_total_cited;
```

	@author_total_cited
	3

  - **find_number_of_publications(email, @total_count)**

This procedure finds the total number of publications that are published by the specified author.

```
set @total_count = 0;
call gokberk_karaca.find_number_of_publications('email1', @total_count);
select @total_count;
```

	@total_count
	3

  - **insert_publication(title, pages, doc_link, s_id)**

This procedure inserts a new publication with correct id, date values. Since we have composite primary key on publication table it is not possible to define auto increment during table creation. To have consistent values we used procedure for publication insertions.

```
call gokberk_karaca.insert_publication('publication from procedure', 72, 'docs.com/how_to_publish', '19');
```

  - **insert_submission(title, doc_link, email)**

This procedure inserts a new submission to the submission table. Again we have composite primary key, procedure increments the id correctly.

```
call gokberk_karaca.insert_submission('submission from procedure', 'docs.com/how_to_submit', 'email1');
```

- **find_most_popular(email, @most_popular_p_id)**

Finds the most popular publications of a given author and return the publication of the most popular publication.

```
call gokberk_karaca.find_most_popular('email1', @most_popular);
```

@most_popular
16

- ## Constraints
  - All foreign key attributes are checked before insertion, deletion and update since they have to be in another table.
- ## Views
  - cited_authors

There is no direct relations between an author and publications that is cited author's any publication. When calculating citation count it is useful to have a view that holds the authors and their publications that are cited author's work.

```
SELECT * FROM gokberk_karaca.authors_cited;
```

email	cited
email1	11
email2	12
email1	13
email1	15
email2	16

- ## Secondary Indices
  - **publication(title)**

Title attribute of publications are used as secondary indices since a user can search publications by their titles on main page.

- **publication(publication_date)**

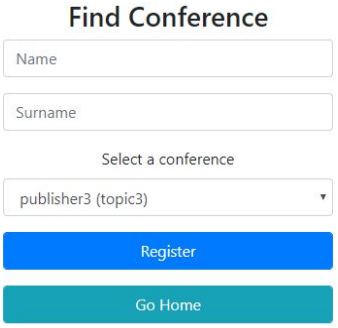
Title attribute of publications are used as secondary indices since a user can do a range search to publications by dates.

6. User’s Manual

At the welcome page, users can choose between signing in, signing up and finding a conference.

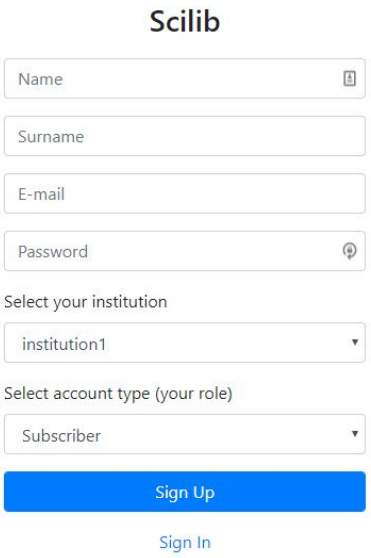
- ## Finding a conference

  Users can choose between conferences that are available for registration. They type their name and surname, choose the conference from dropdown list and register.

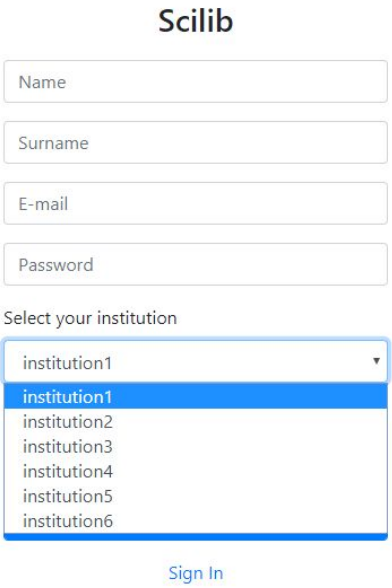


- **Sign Up**

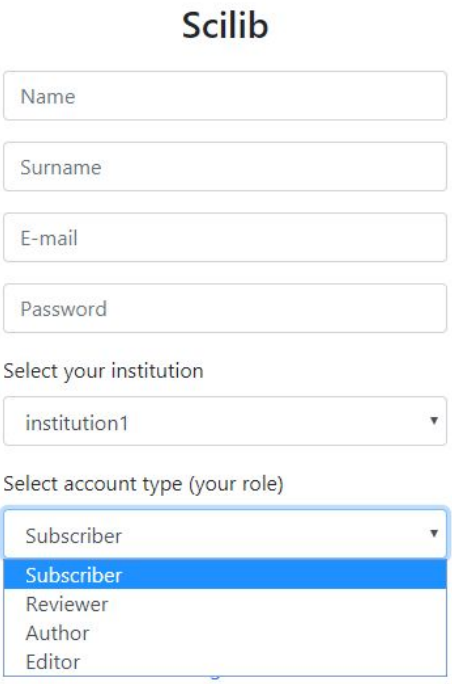
  Upon signing up, users choose their type account. They either become authors, editors or reviewers.
  - Authors can make submissions to different publishers
  - Editors are assigned to different publishers and edit their submissions
  - Reviewers are invited by editors to review the submission



❖ Select institution

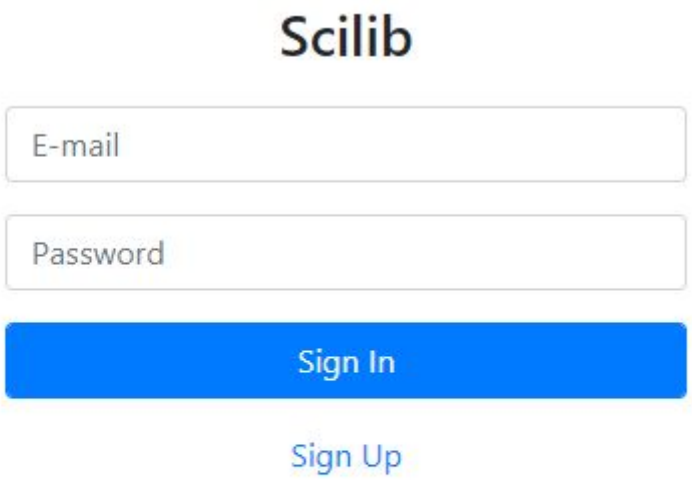


❖ Select user type



- **Log-in**

User can sign- in by using their emails and passwords.



- **Main Page**

  After signing in, the user is redirected to the main page. Here he/she can search for publications according to *publication name, publisher name* or *author name*. When the search is made, people can open the *publication* or *publisher* page. Different User types have different view of main page:

  - Authors can go to their *publications* page or to the *make a submission* page

  - Editors can go to their *edit submissions* page

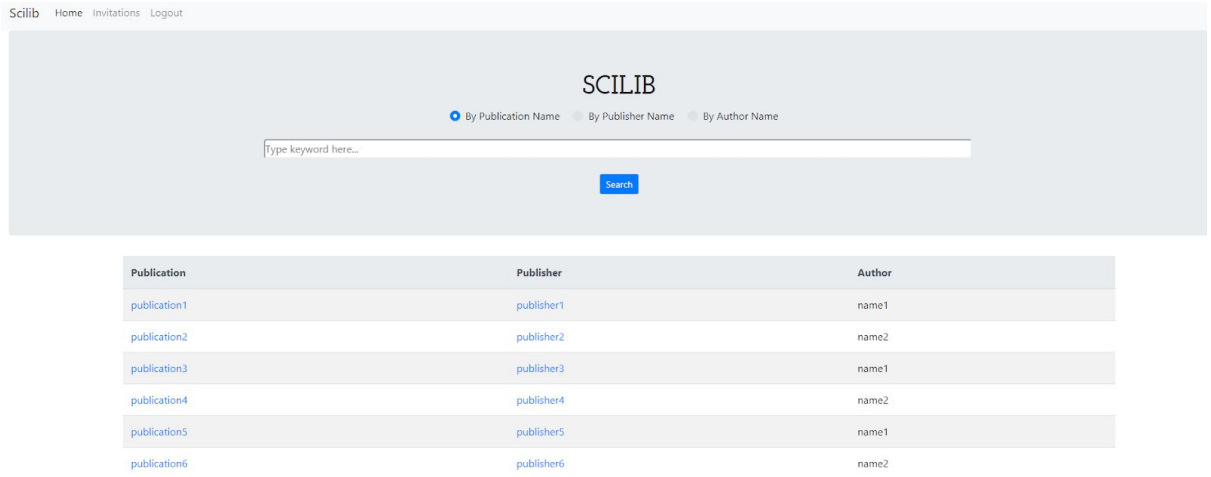
  - Reviewers can go to their *invitations* page to see the invites to review a submission

- ## Search Publications

Search for a publication can be done for 4 filter types:

- By publication name
- By publisher name
- By authors name
- Between Dates

All searches results in a table containing entries for found search results.

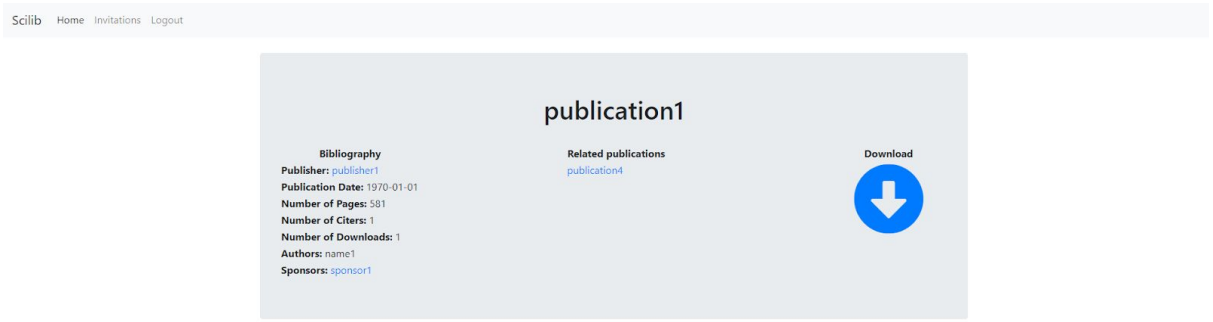


Search for a publisher or an author can be done in one way:

- By name

- **See Publications**

After a search users can see detailed information of a publication, they can download it by going to publication's document link. From this page they can go to the publisher's page, author's page or the pages of other publications who cited the publication.

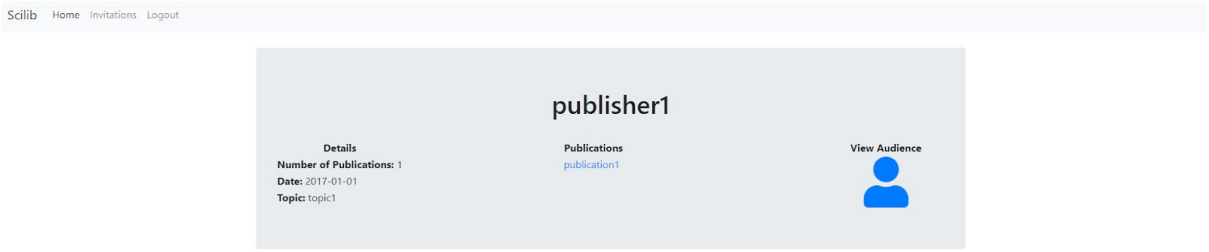


- ## See publishers

After a search or from a related publication users can see details of a publisher. If a publisher is journal user redirects to journal page if a publisher is conference user redirects to conference page.

- ### See Conference Page

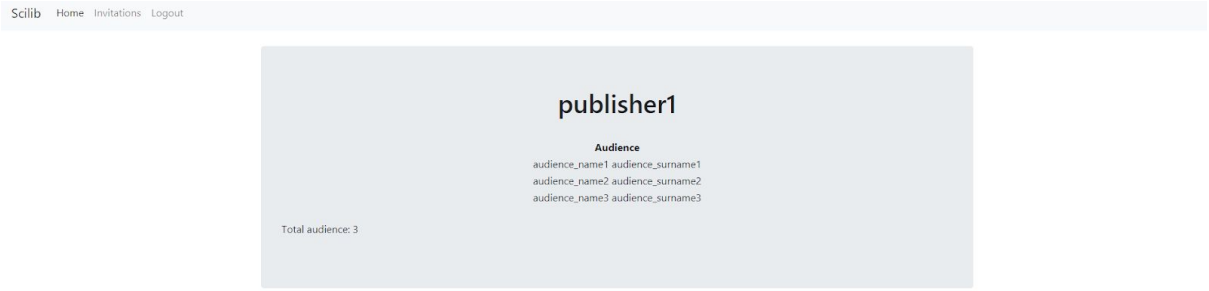
Users can see the details of conference from this page. Whether it is passed or upcoming. Users can see the publications that are published under this conference and go to these publications' pages. If it is a passed conference they can view the audiences of the conference.



,

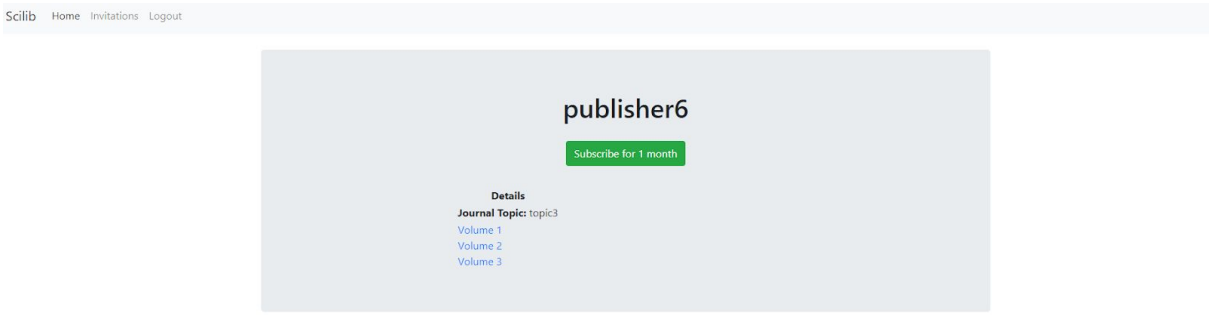
- ### See Audiences

Users can see the audience from this page.

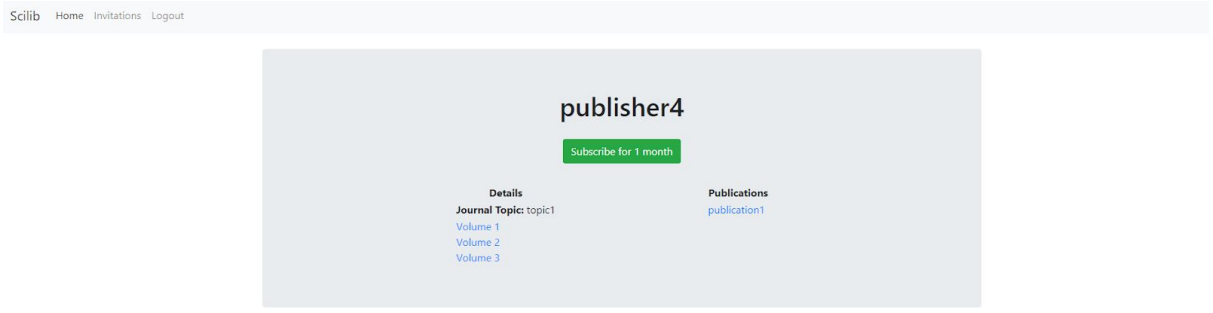


- **See Journal Page**

Users can see the the details and volumes of a journal here. If they click on subscribe button they get subscribed for the particular journal for a month.



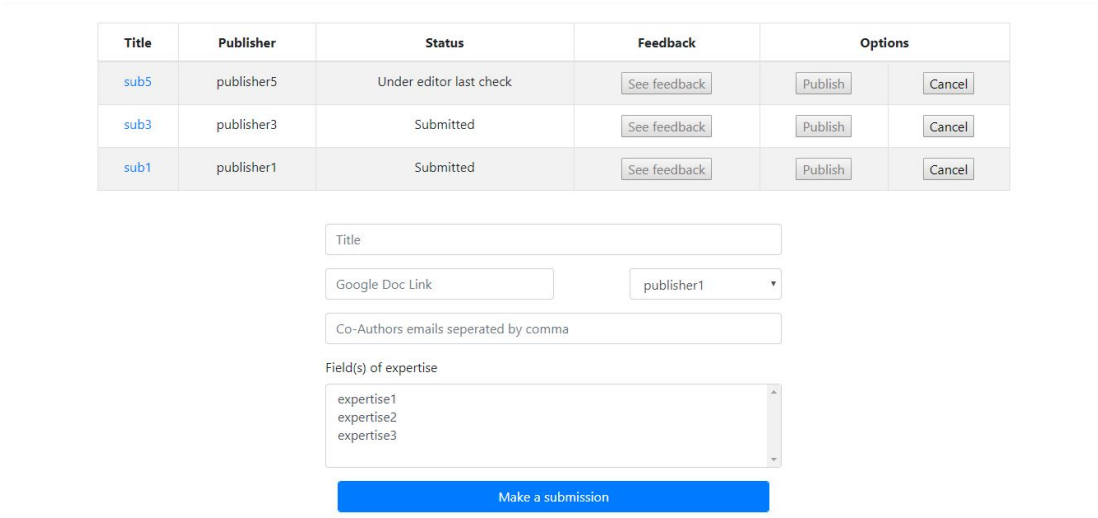
Users can select a volume and see details of volume at the same windows. When clicked on a volume a list which shows the publications under that volume is presented.



- ## Author Publication & Submission Page

  Authors can see their publications in their *publications* page.

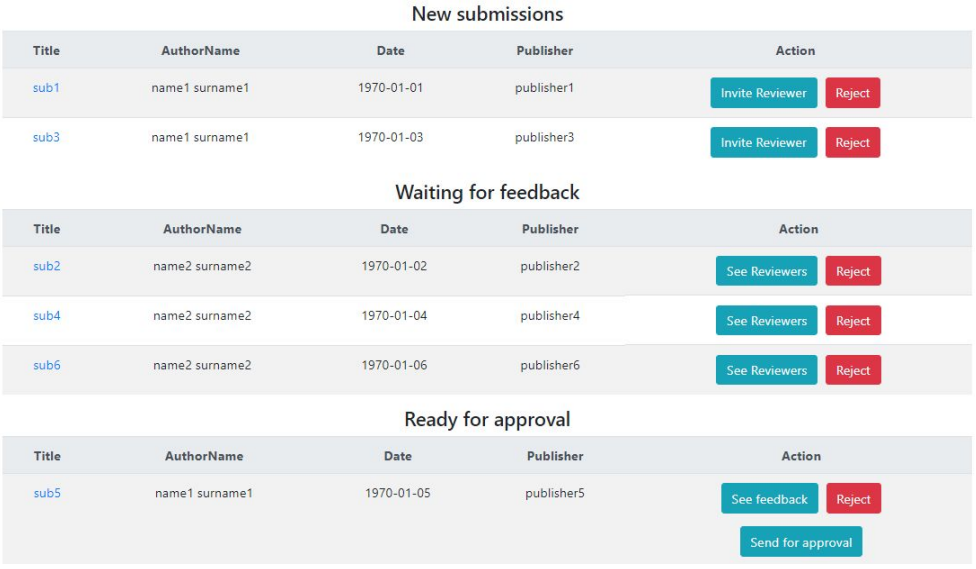
  In the *submissions* page authors can see the status of their status of their submission, the feedback if it has any, cancel the submission or approve and publish if it is ready to be published. Moreover, users can make new submissions here.



- ## Editor Submissions Page

  Editors can see all submissions related to their publisher in their Edit Submissions page. Submissions are separated into 4 categories in this page.
  - *New submissions* are just submitted by authors and do not have reviewers assigned to them. Editors can invite reviewers for the review.
  - *Waiting for feedback* are submissions that have invited reviewers for it. The submission stays here until all of the invited reviewers gave feedback to the submission. Editors can see the reviewers.
  - *Ready for approval,* when all of the invited reviewers gave feedback, the submission comes here. Editors can see the feedbacks and send the submission to the author for the approval if he thinks that the feedback is okay.
  - *Waiting for approval,* after sending the submission to the author, the submission drops here and waits until the author accepts it.

  All of the submissions can be rejected at any time except in waiting for approval stage.



- ## Editor submissions functionality

  If editor presses on the *Invite Reviewer* button, a pop-up window appears. He/she can search reviewers by name and field of expertise and invite them. After inviting reviewers, the submission will be in *Waiting for feedback* stage. If editor presses on see reviewers, a pop-up window appears where the editor can see all of the invited reviewers. If a editor is a editor of an journal, editor can end the volume series.

  Pressing on *See feedbacks* opens a pop-up window with all feedback from reviewers.
  Sending for approval will send the submission to the user to approve for publication
  Reject will delete the submission from the system

- **Reviewer Invitations page**
  Reviewers can see the submissions that are invited to. They can either reject them or give the feedback and send to back to the editor.