Conference Management System – Team Yankee

Dwanika Dakshinamurthy	-	2968218
Girish Sivadanam	-	2501073
Lokesh Kumar Jamjoor Ramachandran	-	2596208
Preneesh Sudhir	-	2742573
Vishwas Bangalore Vijayendra	-	2337250

TECHNISCHE UNIVERSITÄT DARMSTADT

1. **INTRODUCTION** **1**

2. **TECHNICAL DOCUMENTATION** **2**

2.1. SOFTWARE ARCHITECTURE 2

2.2. PROJECT SETUP 3

2.3. RUN PROJECT 3

2.4. PROJECT STRUCTURE 4

2.5. AUTHENTICATION 5

2.5.1. Local Storage: *5*

2.5.2. Authentication header: *5*

2.5.3. Token Secret Key *5*

2.6. USER INTERFACE 6

3. **USER DOCUMENTATION** **7**

3.1. REGISTRATION 7

3.2. LOGIN 7

3.3. USERS DASHBOARD 8

3.4. SUBMISSION 8

3.5. REVIEWS 10

3.6. CHAIR 12

3.7. RESPONSIVE DESIGN 12

3.8. CONCLUSION 13

1. Introduction

The Conference Management System(CMS) is a conference management software, offering a web based solution for managing delegate registration, paper submission and review. It is a comprehensive and powerful, yet easy to use software for online conference management. This system is commonly used for scientific publications and reviews. The CMS software consists of Chairs and Users. The Chair or Administrator manages the conferences. Key duty of a Chair is to assign submission to peers for review and manage deadlines for submissions and reviews. Users register and log into the CMS. Users have two roles here. An User can choose be an author, a reviewer or both. An author can submit a scientific paper, which will be reviewed by a group of peers assigned by Chair. The reviewers will analyze and evaluate the author's research.

In this document, we provide the technical aspect as well as the user documentation. The technical aspect describes the architecture of our application which includes the technologies used for designing the CMS website application. We have given brief explanation of the modules used in our applications. This will be enough for any technical person to take over the project and further add on new functionality. The user documentation briefly describes how to use the application which will be useful for people who want to use the system for creating and managing conferences. In this document, we have not discussed in depth about the technologies used. We have only mentioned the software required to develop or build on our application.

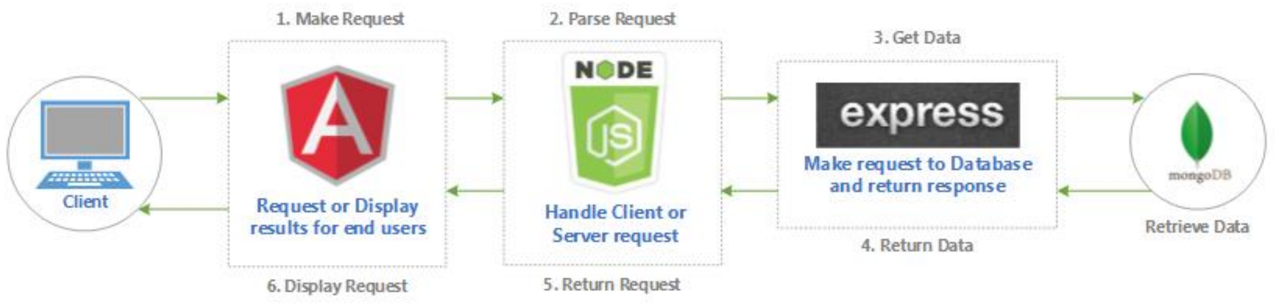
2.Technical Documentation

2.1. Software Architecture

In this project we have used MEAN stack. MongoDB as Database, AngularJS as frontend, , Nodejs as the server with Expressjs used as backend framework. We have used JSON Web Tokens for authentication and token generation and management.

- M **MongoDB**, the world's leading NoSQL database that stores its data into a JSON-like formatted binary file (Binary JSON - BSON). Easy to use and simple to work with JSON.
- E **Express**, a lightweight, minimalist framework built for Node.js for creating web applications, APIs, and HTTP facilities.
- A **AngularJS**, the Model-View-Whatever JS framework makes API consuming simple, optimized for mobile development, same code base for browser, etc.
- N **Node.js,** runs on Chrome's V8 engine and capable of non-blocking, event-driven I/O, handle multiple requests on a single service without them blocking each-other (hence non-blocking).

The following diagram shows the flow of data from frontend to server and server to mongodb



(source: www.optisolbusiness.com)

We used Webstorm as IDE. Which is simple to setup and handles the project with Git integration which makes versioning very easy to use.

2.2. Project Setup

Install the following applications:

Nodejs

Npm

Bower

Mongodb

(* make sure to setup the environment variables for node and mongodb)

Download the project files from git

```
git clone ssh://git@scm.informatik.tu-darmstadt.de/iptk-
ss2016/iptk-ss2016-team-yankee.git
```

The backend dependencies are mentioned in package.json. Install the backend dependencies from npm (from project root directory)

```
npm install
```

The frontend dependencies are mentioned in **bower.json**. Install the dependencies from bower (from project root directory)

```
bower install
```

2.3. Run Project

To run the project first start mongodb service, to do that, run the following command

```
Mongod
```

Then open a new terminal/ command prompt and navigate to **/api** directory.

```
cd api
```

The **server.js** contains the code to start server. Start the server by typing the following command

```
node server.js
```

Now open any web browser and navigate to http://localhost:3000 to see the app.

2.4. Project Structure

```
api/                        <!-- all backend related files-->
    server.js               <!-- starts node application -->
    config.js               <!--stores secret key and token validity time settings-->
    routes/                 <!--backend api routes -->
    schemas/                <!-- mongodb collection schemas -->
    uploads/                <!-- uploaded pdf documents -->

app/                        <!-- all frontend related files -->
    styles/                 <!-- all css files -->
    js/                     <!-- all frontend  java scripts -->
       controllers          <!-- angular controllers -->
       app.js               <!-- angular module definitions and settings -->
       routes.js            <!-- angular routes -->
    img/                    <!-- images and icons -->
    lib/                    <!-- created by bower install, frontend dependencies -->
    views/                  <!-- html pages -->
    index.html              <!--the main html which loads angular and all libraries-->
node_modules/               <!-- created by npm install, backend dependencies -->
.bowerrc                    <!-- tells bower where to put files (public/libs) -->
bower.json                  <!-- tells bower which files we need -->
package.json                <!-- tells npm which packages we need -->
```

```
 mongodb collections
  {
    db_name: "cms",
    collections: [ "users", "submissions", "reviews", "chair" ]
  }
```

We have divided the project into **api** and **app** directories

api directory contains all the backend related code. **app** directory contains all the frontend related code.

In this project we assumed single conference, and the conference name is hardcoded in app.js file as a constant.

```
constant('config', {
  conference_name:"cms"
})
```

2.5. Authentication

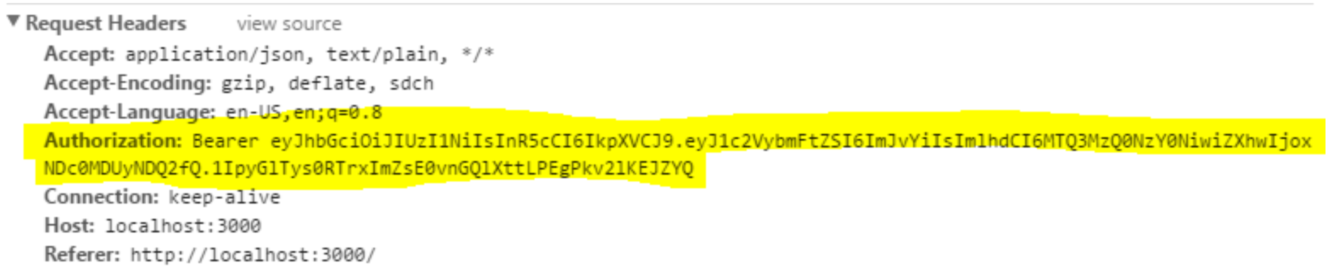
When a new user is registered, a token is generated and signed with the secret key and stored along with user email and password; also sent back to client. The angularjs will save the token and user name in the browsers local storage.

2.5.1. Local Storage:

local storage is a method of storing data in a browser, it's like cookies but more secure and large amounts of data can be stored without affecting website performance. Local storage data is never transferred to server and is unique perdomain, all pages from same origin can access and store same data.

2.5.2. Authentication header:

Every api call should be authenticated, so we send the token as authentication header in every http call. The server checks the token and verifies the validity of the token. If the token is valid then the user request is processed. If the token is invalid then it return 403 forbidden response. The following picture shows how the token is sent to the server



```
Accept: application/json, text/plain, */*
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6ImJvYiIsImlhdCI6MTQ3MzQ0NzY0NiwiZXhwIjox
NDc0MDUyNDQ2fQ.1IpyGlTys0RTrxImZsE0vnGQ1XttLPEgPkv21KEJZYQ
Connection: keep-alive
Host: localhost:3000
Referer: http://localhost:3000/
```

2.5.3. Token Secret Key

The Token is decrypted with the secret key and checks for token validation and expiry data. The secret key and token validation duration is mentioned in api/config.js file

```
module.exports = {
  'secret': 'SMURF',
  'tokenLifeTime': '7 days'
};
```

A token is sent back after login, with validity of 7 days. So for 7 days if the user doesn't logout manually, he can directly login in to the application from the token which is stored in the local storage. After 7 days the token is expired and user have to login again. Also each time a user logs in, his old token is replaced with a newly created token. When a user clicks on logout, the local storage is cleared.

2.6. User Interface

We have used Angular Material Design for User Interface, which is developed and maintained by Google. It supports lots of features and seamless interface designs, icons which gives better user experience. It also supports multiple theming. You can change the theme very easily.

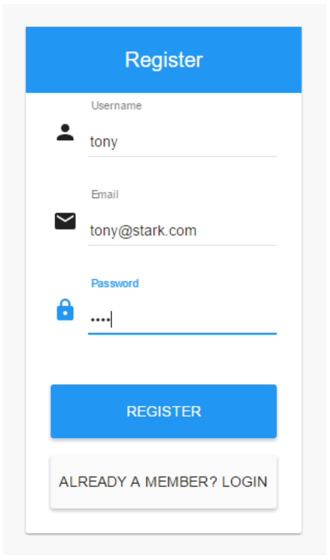
The theming options are mentioned in **app.js** file. You can change color from blue to any other color.

```
.config(function($mdThemingProvider) {
    $mdThemingProvider.theme('default')
      .primaryPalette('blue')
  })
```

3.User Documentation

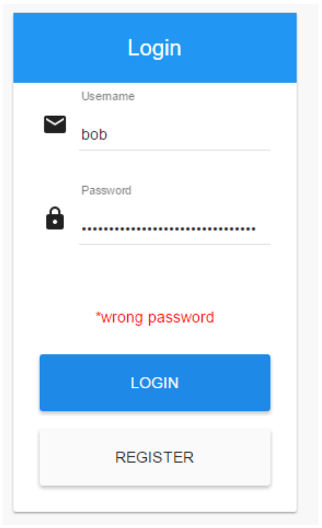
3.1. Registration

Users can register to the application from registration page. A login page is displayed when the application is started. User can select **register** button to navigate to registration page. Which looks like this:



3.2. Login

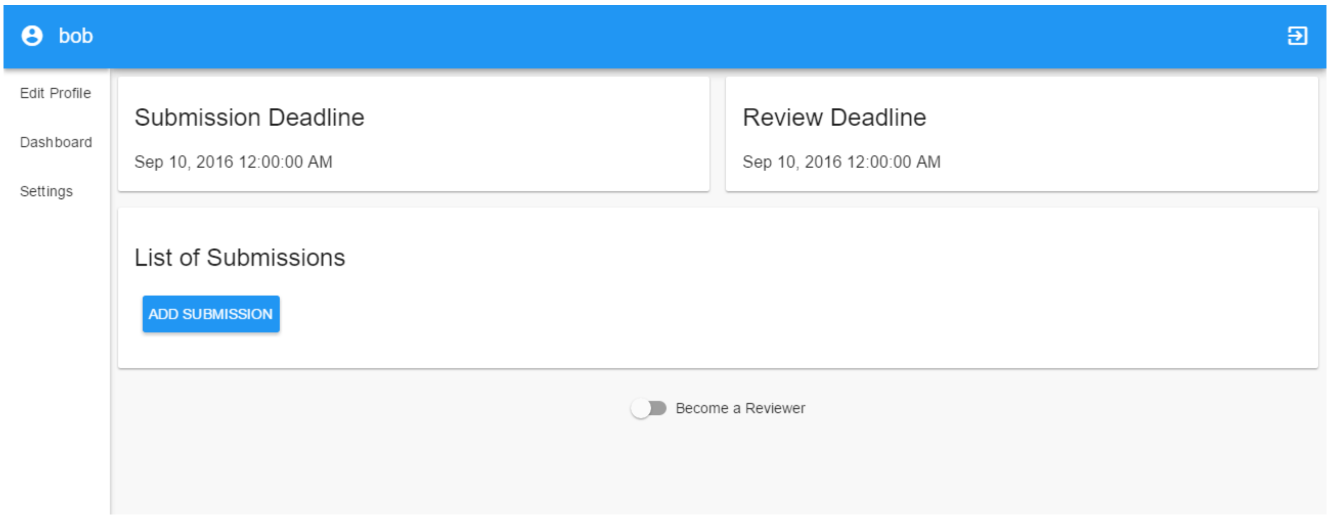
The home page will show the login page. A user can enter his credentials and click login, if the user name doesn't exist or if the password is wrong, an error message is shown, like this.



If the user credentials are correct then he is redirected to user dashboard page.

3.3. Users Dashboard

As soon as user logs in, he is redirected to dashboard which looks like this.



Submission Deadline

Sep 10, 2016 12:00:00 AM

Review Deadline

Sep 10, 2016 12:00:00 AM

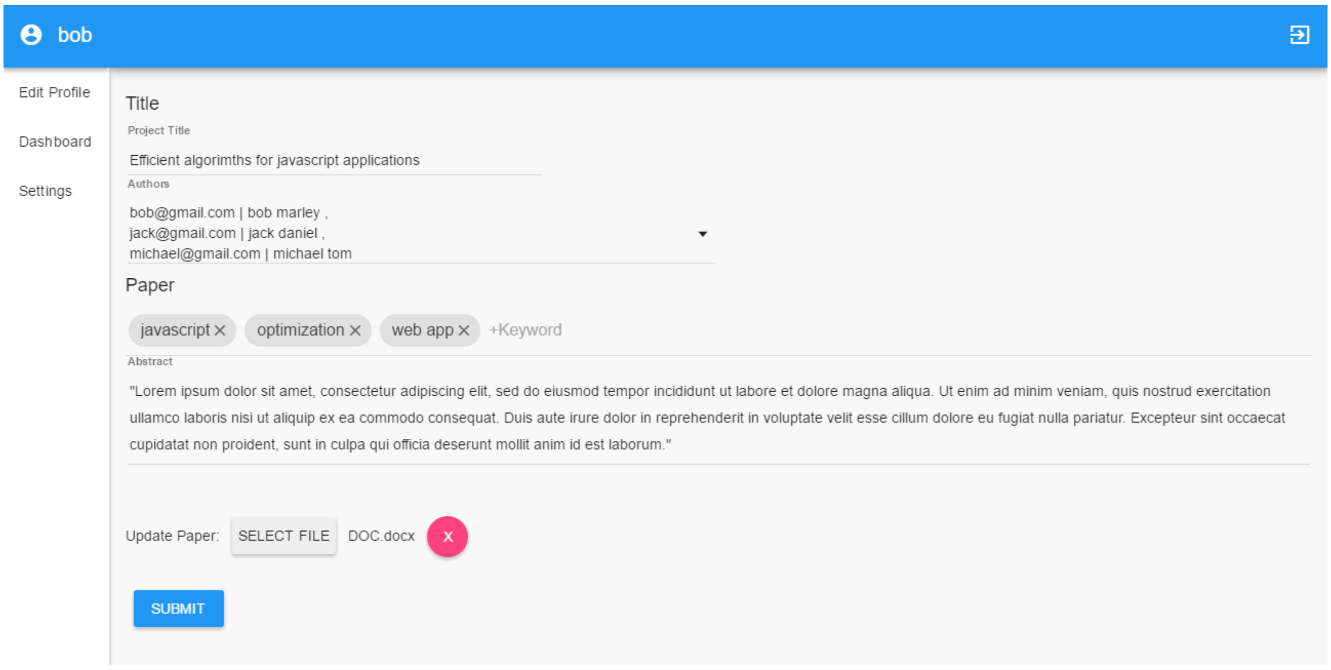
List of Submissions

It shows submission Deadline and Review Deadline on the top. Followed by list of submissions and a button to add submissions.

We assumed every user is an author and he can add submissions. Also a user can become a reviewer by clicking the “Become a Reviewer” button. If a user becomes reviewer then he is added to reviewers list and he can be assigned with other authors submissions.

3.4. Submission

A user can create a submission by clicking “Add Submission” button from the dashboard. Then he is redirected to new submissions page, where he can fill in the details.


bob

Edit Profile

Dashboard

Settings

Title

Project Title

Efficient algorimths for javascript applications

Authors

bob@gmail.com | bob marley ,
jack@gmail.com | jack daniel ,
michael@gmail.com | michael tom

Paper

javascript × optimization × web app × +Keyword

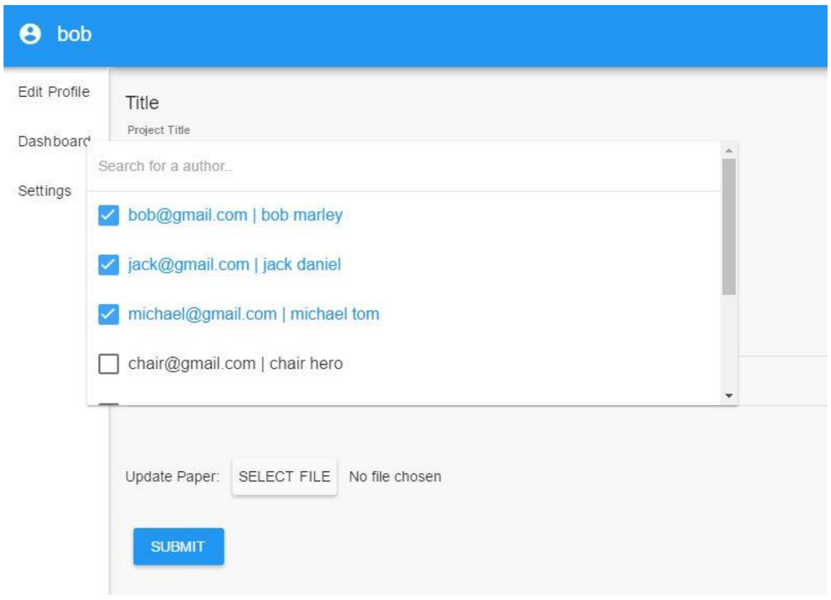
Abstract

"Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

Update Paper: SELECT FILE DOC.docx X

SUBMIT

A use can select multiple co authors like this:


bob

Edit Profile

Dashboard

Settings

Title

Project Title

Search for a author..

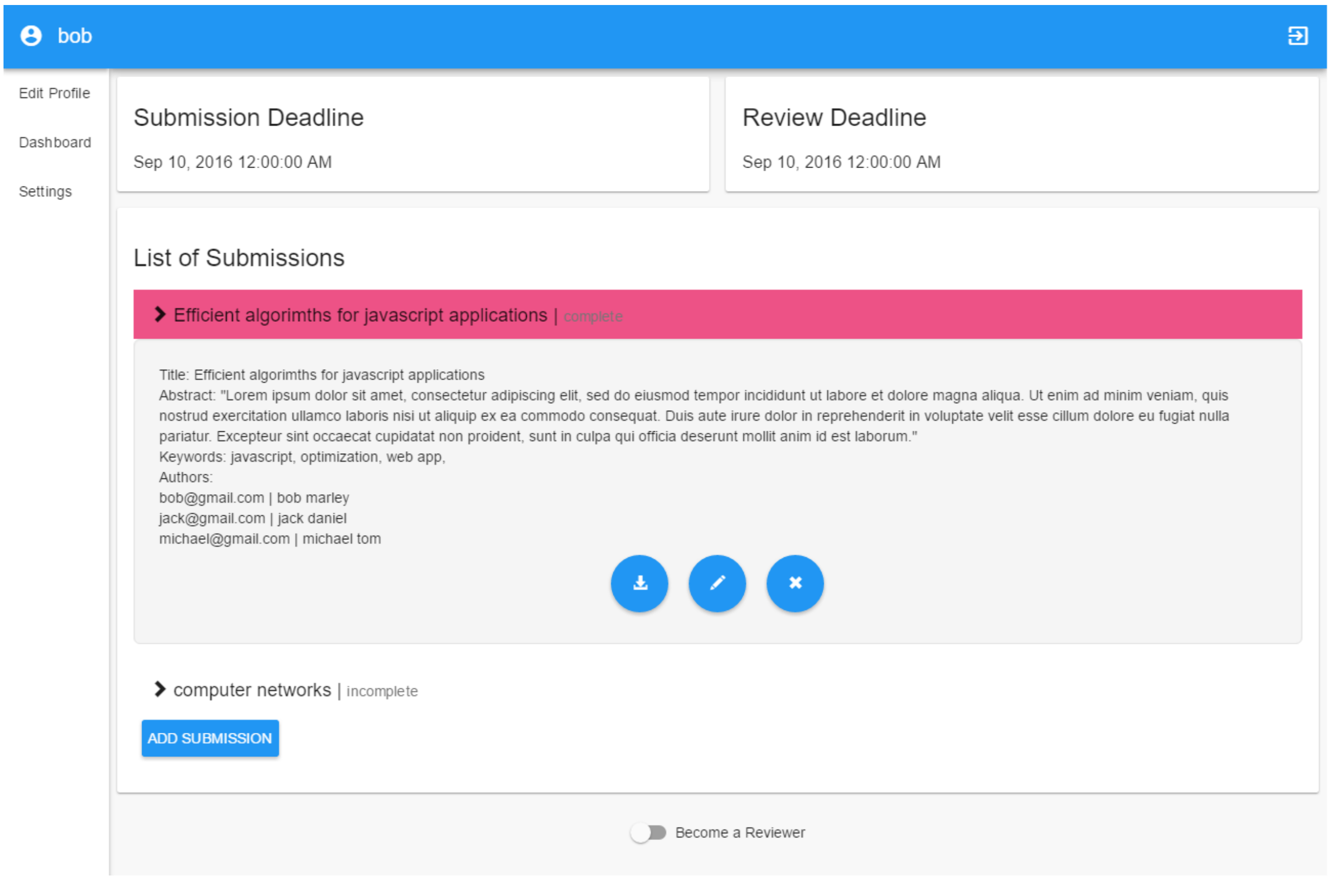
- [x] bob@gmail.com | bob marley
- [x] jack@gmail.com | jack daniel
- [x] michael@gmail.com | michael tom
- [ ] chair@gmail.com | chair hero

Update Paper: SELECT FILE No file chosen

SUBMIT

After clicking submit, the submission is stored in the database, if the submission is incomplete then the status of the submission is assigned with “incomplete”, as soon as user fills all the details of the submission then the submission status is changed to “complete”

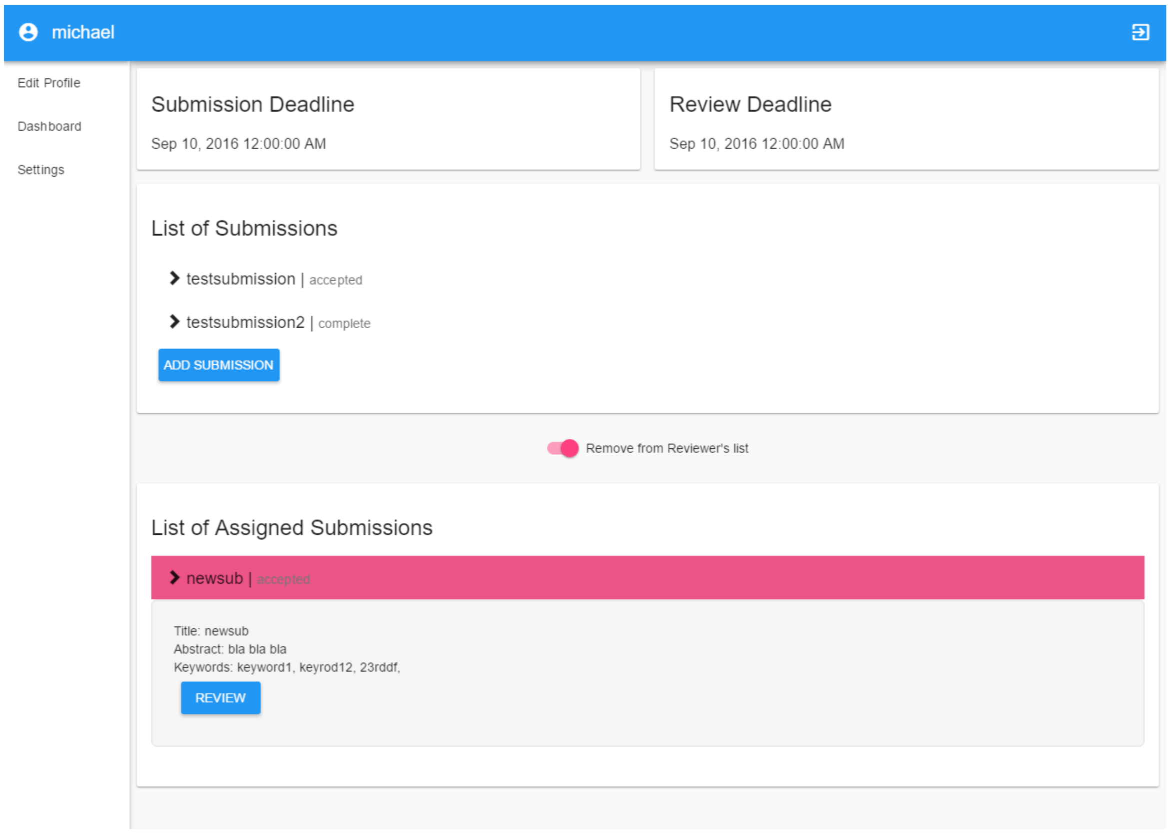
A list of submissions along with the status is displayed in the users dashboard, user can click on the submission which will expand the details of the submission, here he can download the paper, edit the submission or delete the submission by clicking the buttons on submission details



Title: Efficient algorimths for javascript applications
Abstract: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."
Keywords: javascript, optimization, web app,
Authors:
bob@gmail.com | bob marley
jack@gmail.com | jack daniel
michael@gmail.com | michael tom

3.5. Reviews

If a user becomes a reviewer then he will be assigned with other users submissions. A list of assigned submissions is shown in the dashboard. A user can click on the submission to view details about the submission, which looks like the following figure.



Submission Deadline
Sep 10, 2016 12:00:00 AM

Review Deadline
Sep 10, 2016 12:00:00 AM

List of Submissions

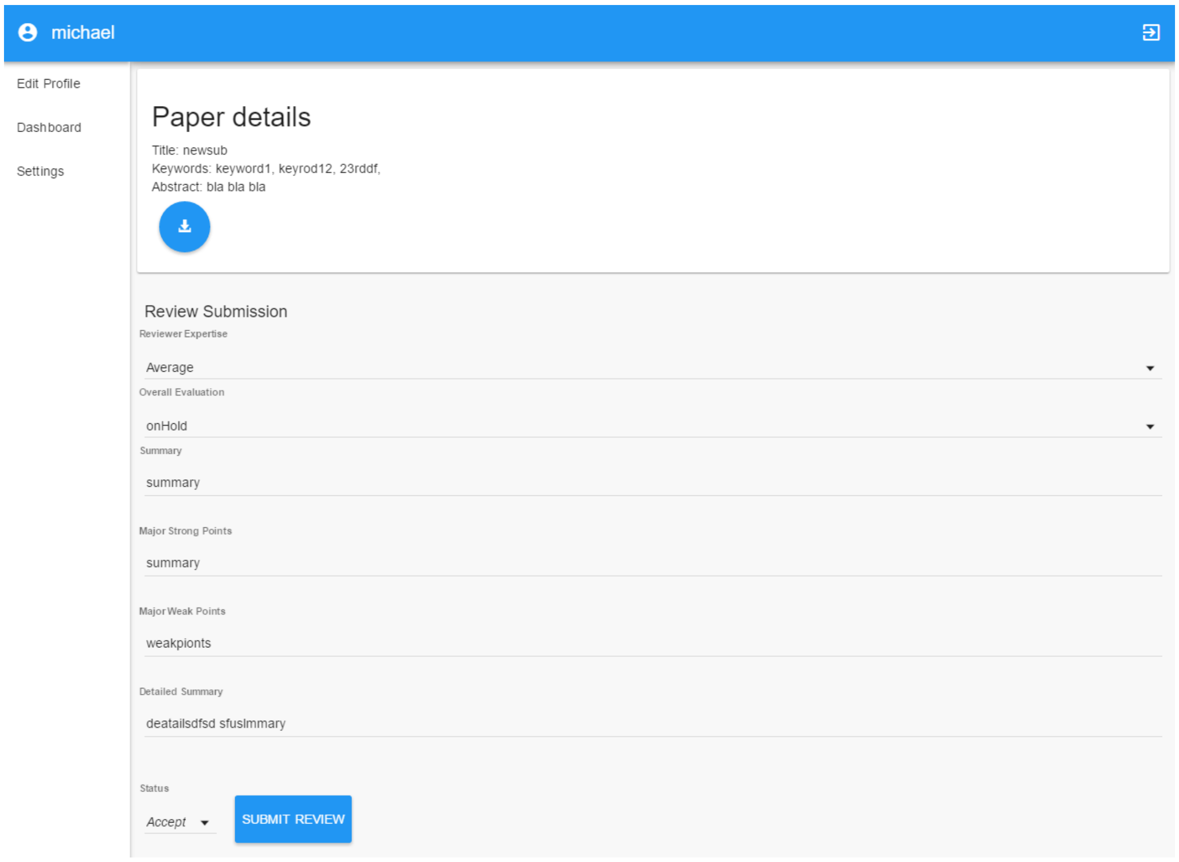
- testsubmission | accepted
- testsubmission2 | complete

List of Assigned Submissions

- newsub | accepted

Title: newsub
Abstract: bla bla bla
Keywords: keyword1, keyrod12, 23rddf,

The reviewer can click on "review" button to review a paper, then it will take him to review page, which looks like this



Paper details

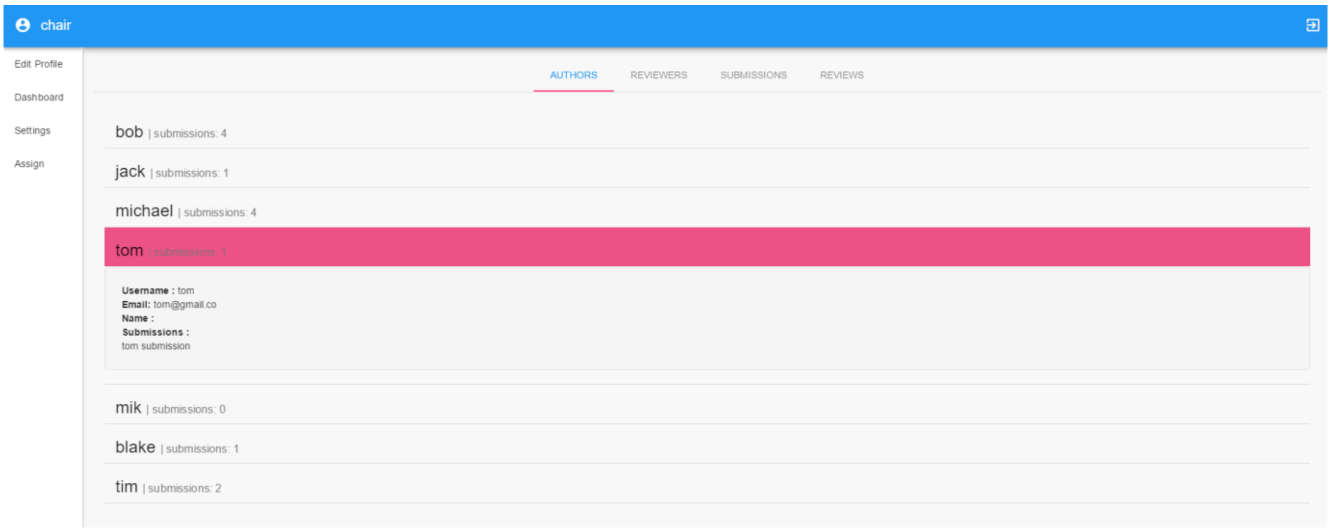
Title: newsub
Keywords: keyword1, keyrod12, 23rddf,
Abstract: bla bla bla

Review Submission

Reviewer Expertise: Average
Overall Evaluation: onHold
Summary: summary
Major Strong Points: summary
Major Weak Points: weakpionts
Detailed Summary: deatailsdfsd sfuslmmary
Status: Accept

3.6. Chair

The chair dashboard looks like this.



bob | submissions: 4

jack | submissions: 1

michael | submissions: 4

tom | submissions: 1

Username : tom
Email: tom@gmail.co
Name :
Submissions :
tom submission

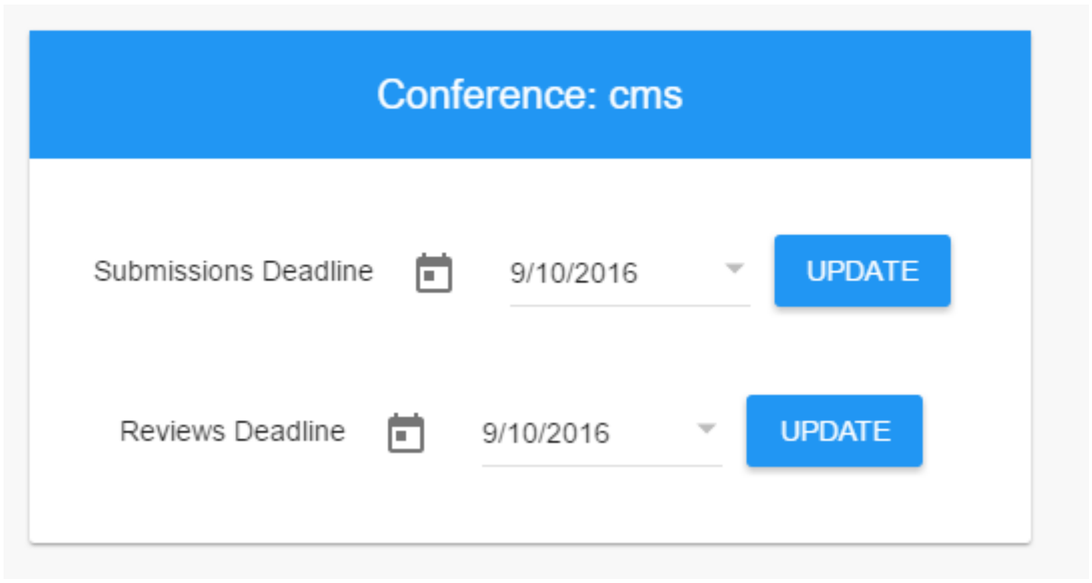
mik | submissions: 0

blake | submissions: 1

tim | submissions: 2

The chair dashboard lists all the list of authors, reviewers, submission and reviews.

He can set the submission and review deadlines in the settings page, which looks like this

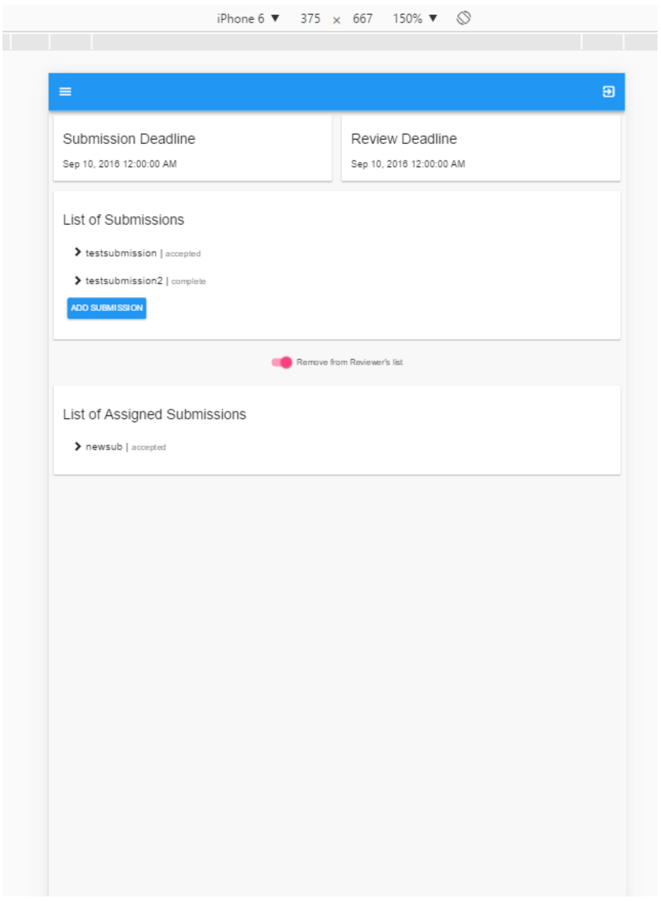


Submissions Deadline 9/10/2016

Reviews Deadline 9/10/2016

3.7. Responsive Design

The side nav bar is hidden in mobile devices and can be seen by clicking the top left menu button



Submission Deadline
Sep 10, 2016 12:00:00 AM

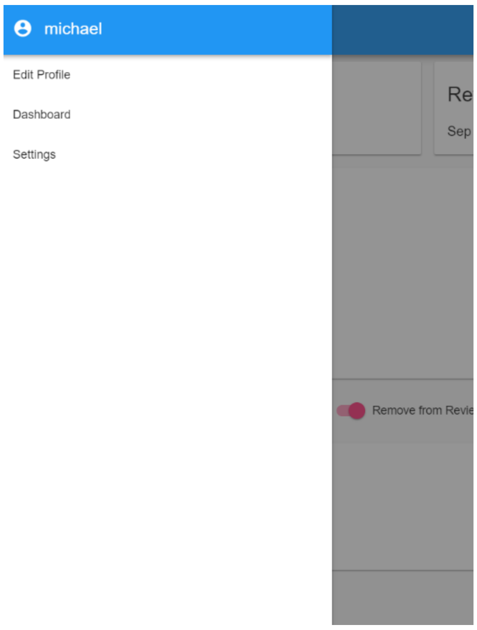
Review Deadline
Sep 10, 2016 12:00:00 AM

List of Submissions
- testsubmission | accepted
- testsubmission2 | complete

ADD SUBMISSION

Remove from Reviewer's list

List of Assigned Submissions
- newsub | accepted



michael

Edit Profile

Dashboard

Settings

Re

Sep

Remove from Revie

3.8. Conclusion

We have designed a very secure, clean and intuitive web application which is very flexible to extend and modify further as per user requirements.

The only feature we hav’t implemented is showing charts and statistics on chair dashboard. Apart from this we have completed every feature from feature list in the backlog.

Further development:
All the bonus features can be implemented easily from our project.