Google Summer of Code

Georgia Tech

Improvement in EvalAI Frontend

CLOUD CV

Your_name
(Your_email_id)

Introduction

My name is Your_name. I am an undergraduate student at Your_Collge_Name . I have been into Web Development for about 3 years now, which has helped me to build a strong foundation. I am familiar with Angular 7, MongoDB and Node.js, along with which I also acquired experience in unit testing using Karma and Jasmine.
Apart from Frontend Web Development, I have been introduced to the basics of some popular programming languages like Python, JavaScript, TypeScript, C and C++.

My goal is to become a full-stack developer, capable of making useful and elegant websites.

Current Education

I am in the third year of college, pursuing a B.Tech. degree in Computer Science at Your_Collge_Name

Contact Information

- Name - Your_name
- Email - Your_email
- Github - Your_Github
- Blog - Your_blog
- Time Zone - Indian Standard Time (IST), +5:30 UTC

Motivation

Why GSOC?

Google Summer Of Code provides a great opportunity to work with amazing people and being a learner I would love to spend my summer experiencing and learning new things every day. Moreover, I would also be able to be a part of a network and organization consisting of great

minded people which would be really crucial in beginning my career and later on in life.

Why CloudCV?

First of all, I found the idea behind this platform to be quite innovative and interesting. I have chosen this project because of my interest and have continuously contributed to this organization since the last 3 months, so I am already acquainted and comfortable with its codebase and the mentors of this organization. Along with this, I am already in contact with many of the contributors and mentors of this organisation so I feel quite comfortable while working, helping other contributors and getting help from them.

Why this idea?

I love working on frontend and I'm also experienced with Angular 7 as I have been working with it since the last 3 years. So I'll be able to give better input on the idea "Improvement in EvalAI Frontend " under GSOC 2020 ideas. It will also help me to further improve my knowledge in Angular 7, materialize css and unit tests along with introducing me with the other important modules and better approach of problem solving. This idea will eventually help me to grow better as a frontend developer and so i choose this.

My Expectations?

I expect to get good mentorship on this project along with proper guidance wherever necessary. I also expect to get help from mentors in certain situations where I may get stuck along with getting continuous feedback and suggestions for improvement on my work in terms of efficiency and productivity. This would help me to give my best in this project.

nithyanithin210@gmail.com

Learning Expectations?

I expect to get outstanding work experience in an open source project. The guidance from an expert and experienced mentor would not only help me grow as a professional, but would also be beneficial in acquiring insights and perspectives on the domain of skills required for this idea. Along with all this, I am also expecting to take my knowledge to another level of efficiency through this idea by getting experience on problem solving, feature adding, clean coding etc.

Experiences

I have been working with MEAN(MongoDB, Express,Angular, Nodejs) stack for 3 years from now. I have made 4 personal projects and 2 projects under an internship based on the MEAN Stack. I have worked with AngularJS, Angular2 and Angular 7 as well.

Following are the list of personal projects i have done:

CloudCV Contributions

I have made a few contributions in cloudCV, these are:

CloudCV/EvalAI:

CloudCV/EvalAI-ngx:

Other Open Source Contributions:

nithyanithin210@gmail.com

Project Description

Project Introduction

The project that I would like to work on for **CloudCV** for GSoC'20 is "**Improvement in EvalAI Frontend**".
This project involves fixing the last remaining kinks in the UI and incorporating latest UI as per the feedback received from the challenge hosts and participants of the AI challenges organized. It also involves shifting to the latest Angular version, to experience new features and enhancements it came with, by shifting the current codebase from Angular 1 to Angular 7 along with improving the test coverage, refactoring and focussing on making it robust and modular.

Project Goals

The Frontend to be made should have the following features -

- ❏ To participate in a challenge based on an invitation by the challenge host.
- ❏ To have testimonials, statistics added on the home page.
- ❏ To have enhanced the UI for displaying the submissions.
- ❏ To have the feature for prettier URLs for the challenge
- ❏ To have support for editing challenge details.
- ❏ To have the CRUD feature to challenge hosts on submissions.
- ❏ To have pages for maintenance, 400, 500, 404 errors on the website.
- ❏ To have the feature for filtering the challenges on UI
- ❏ To export submissions in different formats by challenge hosts for analysis.
- ❏ To have service workers added for making the website loading faster.
- ❏ To be modular and robust
- ❏ To be test covered
- ❏ To be refactored

nithyanithin210@gmail.com

Extended Project Goals

- ❏ Add lazy loading for further improving the performance.
- ❏ To separate the main app-routing-file.
- ❏ Solve the following issue found while working on this project idea:
  - ❏ EvalAI-ngx issue #309 [Remove banned email Id button not working correctly]
  - ❏ EvalAI-ngx issues #310 [View all submission Tab redirecting to Participant Tab]

  Along with these mentioned issues, all the other issues that are required to be solved for smooth functionality of the website.

Project Plan

My Approach

- ❏ **Participate in a challenge based on an invitation by the challenge host.**

  - ❏ Currently, the challenge host can't invite participants to participate in his/her hosted challenges. This feature will allow the challenge host to invite any participant to participate in the challenge.
  - ❏ I will add "Invitation to participate in this Challenge" section in the "challenge settings", just above the "banned email" section. It will allow the host to invite the participants to participate in his/her challenge using the participants email Ids.
  - ❏ I will be taking input of the email in the form of the MatChipInputEvent from the host of the challenge and

nithyanithin210@gmail.com

use it to call the event handling function which will finally call the API.

Function for the handling MatChipInput event:

```
/**
  * Add invite email chip
  * @param event current event
*/
 invite(event: MatChipInputEvent): void {
   const SELF = this;
   const input = event.input;
   const value = event.value;
   SELF.isValidationError = false;
   SELF.message = '';

   if ((value || '').trim()) {
     if (!SELF.validateEmail(value.trim())) {
       SELF.isValidationError = true;
       SELF.message = 'Please enter a valid email!';
     } else {
       SELF.invitedEmailIds.push(value.trim());
     }
   }

   // Reset the input value
   if (input && !SELF.isValidationError) {
     input.value = '';
   }
 }
```

- ❏ I will be using invite users to challenge api written in the EvalAI backend for adding this feature.
- ❏ This API will require certain changes like:

- ❏ Modifying the condition for sending the invitation email as mentioned here.
- ❏ Add valid_EmailIds in response so that it can be used to display the list of Email Ids invited till then in the host setting tab.

Hitting the API to send the invitation to the user:

```
/**
  * Updating InvitedEmailIds list
  */
updateInvitedEmailList() {
   const SELF = this;
   const BODY = JSON.stringify({
     data: {
       'emails': SELF.invitedEmailIds
     }
   });
   SELF.apiService.postUrl(

SELF.endpointsService.inviteParticipanttoChallenegURL(SELF.c
hallenge.id),
     BODY
   ).subscribe(
     data => {
       SELF.challenge.emails = data.emails;
       SELF.isInvitedEmailInputVisible = false;
       SELF.globalService.showToast('success', 'Invited
participant emails are successfully updated!', 5);
       this.formerInvitedEmailIds =
this.invitedEmailIds.concat(); // Creating deep copy
     },
     err => {
       SELF.globalService.handleApiError(err, true);
       SELF.globalService.showToast('error', err);
     },
     () => { }
```

```
  );
}
```

- ❏ Invited Participants will receive a notification on their dashboard, just below the section showing count of ongoing challenges, host teams and participant teams, regarding the invitation. The notification will contain the proper required information along with the necessary links, name of the host etc.
- ❏ The rough UI of invitation received by the participant is Invitation_received_by_participant
- ❏ Invited participants will be allowed to accept the invitation using accept_challenge_invitation api written in the EvalAI backend.
- ❏ For getting more idea about the approach of inviting the participant to the challenge, please see EvalAI-ngx #311.

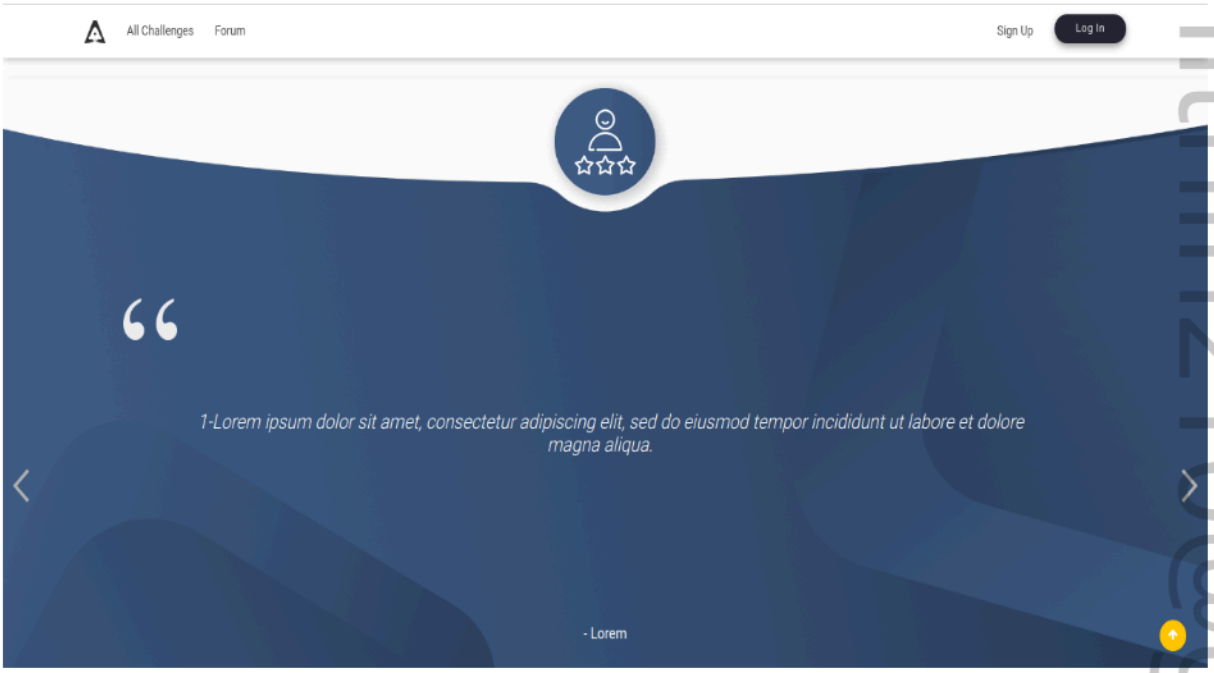
❏ Add testimonials and statistics on the home page

- ❏ Currently, the home page doesn't contain any testimonials or statistics. Under this feature, I will allow dynamic display of statistics and most recent feedback on the home page.
- ❏ I will add all the testimonials in testimonials.component.html. And will use it's selector tag to view it on the home page as just above the "Cite our work" section.

```
/**
  * Added Testimonials
 */
```

```
 <section class="ev-container ev-details text-med-black
ev-super-light-bg">
     <app-testimonials></app-testimonials>
   </section>
```

After using the current testimonials.component.html to adding it in the home page as described above, i got the output as shown below :



Currently, only three hard-coded data is displayed here. I will change it to be fetched from the database directly.

- ❏ Along with this, I will also add a section that will show the statistics based on the feedback received from the challenge hosts and participants.
- ❏ The rough UI of the home page after making these changes is home page UI with testimonials and statistics added

❏ **Enhanced the UI for displaying the submissions.**

- ❏ In this part, I will mainly focus on improving the UI for displaying the submissions as per the provided demo along with some minor enhancement if required.
- ❏ In this part I'll be mainly working with materialize css, html and Typescript to get the UI as expected.
- ❏ Along with enhancing UI, this feature will also require the addition of missing features for the host submission tab like:
  - ❏ Adding leaderboard precision value
  - ❏ Allowing hosts to remove the Email Ids from banned Emails list as mentioned here.
  - ❏ Fix the image display so that challenge's image should be displayed under the my_hosted_Challenges tab.
  - ❏ Allowing the hosts to navigate to the "view_all_submissions" tab as mentioned here.

❏ Prettier URLs for the challenge

- ❏ Currently the format of the url is website_url/challenges/challenge-page/id/overview. It includes Primary key i.e id in the url.
- ❏ In this feature, I will focus on replacing the current url with the unique slug based urls using pipe.

```
// pipe for getting sluggish url
@Pipe({
   name:'UrlPipe'
})
export class UrlPipe implements PipeTransform {
   transform(value:string): string{
       return function(value) {
 if (!value)
 return;
```

```
// make lower case and trim
var slug = value.toLowerCase().trim();

// replace invalid chars with spaces
slug = slug.replace(/[^a-z0-9\s-]/g, '');

// replace multiple spaces or hyphens with a single hyphen
slug = slug.replace(/[\s-]+/g, '-');

return slug;
};
   }
}
```

- I have shown here the basic structure of the pipe I'll use for this purpose. I will also add some other conditions in the function of the pipe if required in order to get the shorter, cleaner and well defined urls.

Support for editing challenge details.

- In this part, I will add the feature of allowing challenge hosts to edit all of their hosted challenge details.
- For adding this feature, I will be using the challenge_detail api written in the EvalAI backend.

```
/**
  * Edit maximum concurrent submission of the challenge
  */
 editMaximumConcurrentSubmission() {
   const SELF = this;
   SELF.apiCall = (params) => {
     const BODY = JSON.stringify(params);
     console.log(BODY);
```

```
      SELF.apiService.patchUrl(

SELF.endpointsService.editChallengeDetailsURL(SELF.challeng
e.creator.id, SELF.challenge.id),
        BODY
      ).subscribe(
        data => {

SELF.challenge.maximum_concurrent_submissions=data.evaluati
on_details;
          console.log(data);
          SELF.globalService.showToast('success', 'The
maximum concurrent submission is successfully updated!',
5);
        },
        err => {
          SELF.globalService.handleApiError(err, true);
          SELF.globalService.showToast('error', err);
        },
        () =>
console.log('EDIT-MAXIMUM-CONCURRENT-SUBMISSION-FINISHED');
      );
    };
```

- ❏ The above function is written for editing the maximum_concurrent_submission. I will write similar functions for the missing editing features of the host.

❏ CRUD feature to challenge hosts on submissions.

- ❏ Currently, the host is only allowed to view and update the submissions with the provided submission-edit-function using the download_all_submission api. In this part, I will add the feature of allowing challenge hosts to delete the submissions.

```
 /**
  * Display delete Submission Modal.
  * @param Submission being deleted
    */
    deleteSubmission(submission) {
    const SELF = this;
    SELF.apiCall = () => {
         const BODY = JSON.stringify({});
         SELF.apiService.postUrl(

SELF.endpointsService.challengeSubmissionUpdateURL(SELF
.challenge.id, submission.challenge_phase,
submission.id),
         BODY
         ).subscribe(
         data => {
         // Success Message in data.message after
deletion
         SELF.globalService.showToast('success', 'Data
deleted successfully', 5);
         },
         err => {
              SELF.globalService.handleApiError(err,
true);
SELF.globalService.showToast('error', err);
         },
         () =>
console.log('SUBMISSION-DELETE-FINISHED')
         );
    };
    const PARAMS = {
              title: 'Delete Submission',
              content: '',
              confirm: 'I understand consequences,
```

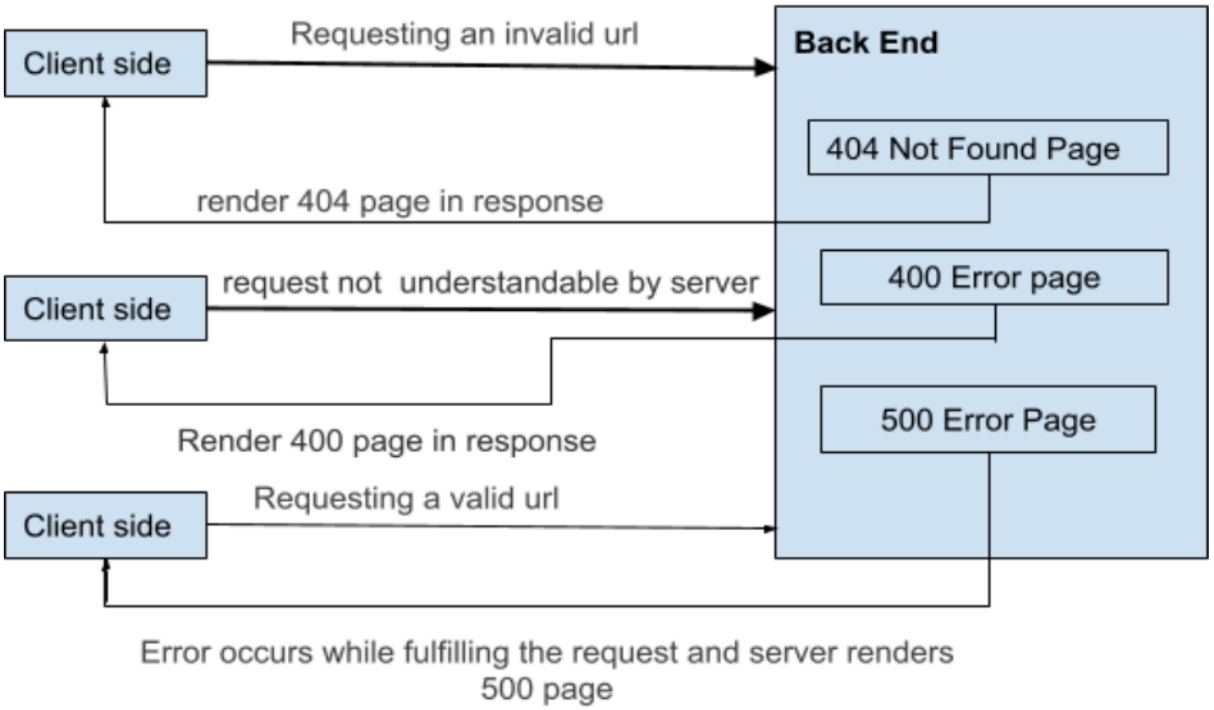
```
delete the submission',
                deny: 'Cancel',
                form: [
                {
                Name: 'submissinDeleteInput',
                isRequired: true,
                label: ' ',
                placeholder: 'Please type in the name of
the submission to confirm',
                type: 'text',
                value: submission[''],
                },
            ],
            confirmCallback: SELF.apiCall
        };
         SELF.globalService.showModal(PARAMS);
        }
```

- ❏ Above code snippet is of the function that will allow the hosts to remove unacceptable submissions. And hence hosts will get CRUD privileges on the submissions.

❏ Pages for maintenance, 400, 500, 404 errors on the website.

- ❏ Currently, the website has only got 404-not-found-page which needs some enhancement in its UI.
- ❏ It doesn't have other pages for maintenance. Under this feature, I will make the pages for 400, 500 and enhance the 404 page of the site.
- ❏ The 404-not-found-page error page will be displayed to the user, when they try to visit a page that doesn't exist at that point of time.
- ❏ The 400 error page will be displayed to the user when they make any bad request to the server.

- ❏ The 500 error page will be displayed to the user when they encounter any “internal server error”.



Overview of how and when the different error pages will be rendered

❏ Filtering the challenges on UI

- ❏ Currently, all the challenges i.e ongoing, upcoming, past are kept on the same page which makes it inconvenient for the users to find a particular challenge.
- ❏ This feature will allow the users to filter challenges based on the names of the challenges and host team.

- In this feature, I will add an API that will filter the challenges based on the title. For this, I will make the following changes in challenge-folder.

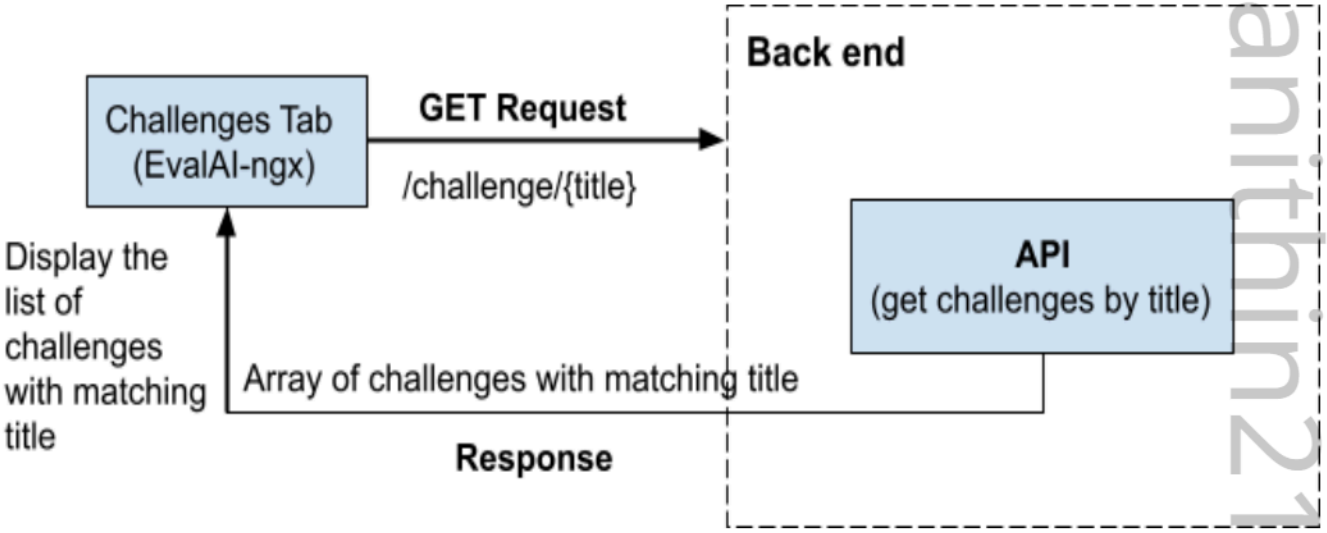
In urls.py:

```
    url(
        r"^challenge/(?P<challenge_title>[\w\s]+)/$",
        views.get_challenge_by_title,
        name = "get_challenge_by_title",
    )
```

In views.py:

```
@api_view(["GET"])
@throttle_classes([AnonRateThrottle])
def get_challenge_by_title(request, challenge_title):
    """
    Returns a particular challenge by their title.
    """
    challenges = Challenge.objects.filter(
        title__icontains=challenge_title,
        approved_by_admin=True,
        published=True
    )
    response = []
    for challenge in challenges:
        serializer = ChallengeSerializer(
            challenge, context={"request": request}
        )
        response.append(serializer.data)
    if len(response) == 0:
```

```
        exception_msg = {"error": "No Challenge found!"}
        return Response(exception_msg,
status=status.HTTP_406_NOT_ACCEPTABLE)
    return Response(response, status=status.HTTP_200_OK)
```



Overview of the API for filtering challenges

- This API will then be hit from front end to get the challenge based on their title by following function:

```
/**
  * Fetch challenge by title
  */
 getChallengebyTitle(challenge_title, callback = null) {
 const API_PATH = 'challenges/challenge/' + challenge_title
+ '/';
   const SELF = this;
   this.apiService.getUrl(API_PATH).subscribe(
     data => {
       if (callback) {
```

```
          callback(data);
        } else {
          SELF.changeCurrentChallenge(data);
        }
      },
      err => {
        SELF.globalService.handleApiError(err, false);
      },
      () => {
        console.log('Challenge', challenge_title,
'fetched!');
      }
    );
  }
```

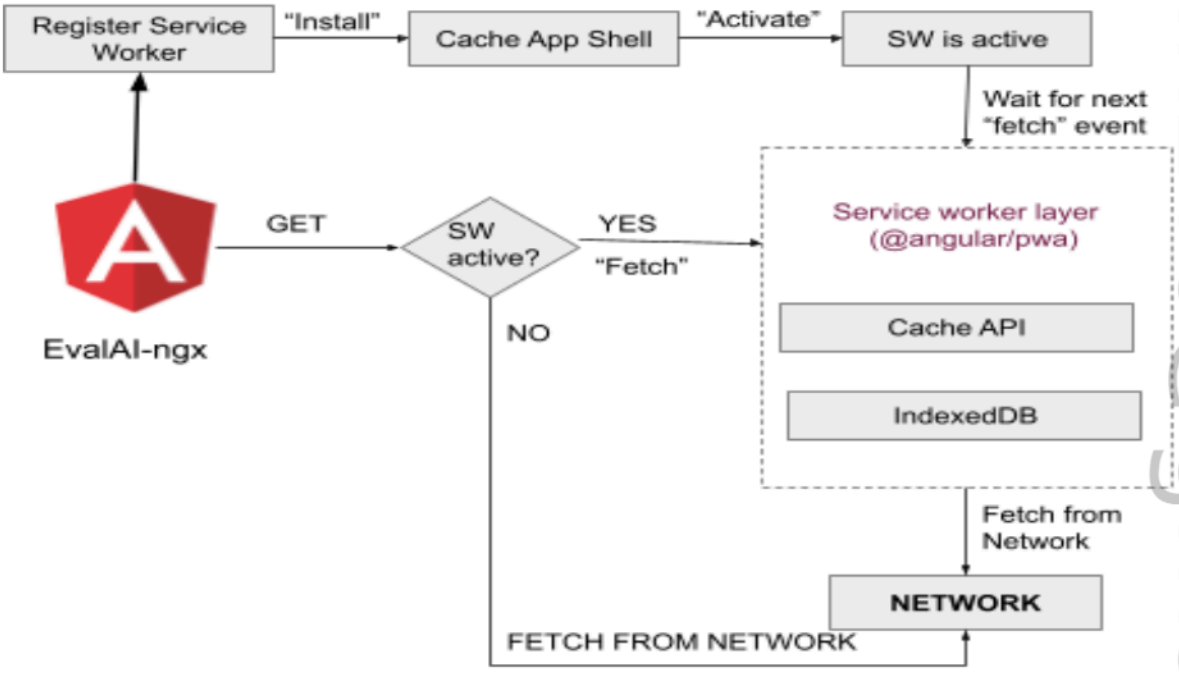
- This feature can further be enhanced to allow users to filter the challenges based on the starting date, ending date etc.

❑ Export submissions in different formats by challenge hosts for analysis.

- Currently, the submission data is only sent in the 'csv' format under my_submissions and view_all_submissions as you can see here.
- In this feature, I will add a function to export the submissions in the 'json' format, 'zip' format and other formats as required for the analysis.
- As the API is written for downloading and storing '.csv' file here, in the similar way, I will write for 'json' and 'zip' files.
- For storing in 'zip' file format, I will be using the zipfile module.
- I will modify the download_all_submissions API for allowing the download in .zip file also so that it can download and be used for analysis.

❑ Add service workers for making the website loading faster.

- ❑ Service workers function as network proxy that intercept all outgoing HTTP requests made by the application and can choose how to respond to them. Below is the diagram through which i tried to show how service workers work and helps to improve the performance of the EvalAI-ngx.
- ❑ I will be using the third party package named "@angular/pwa" for this purpose. This package is provided by angular itself.



Service worker improving the performance

- ❑ Adding this package adds some new files like ngsw-config.json along with making some changes in the app module.
- ❑ I will configure service workers for API calls by using Data Group property in the ngsw-config.json file. The format of adding Data Group is shown below.

```
export interface DataGroup {
                name: string;
                urls: string[];
                version?: number;
                cacheConfig: {
                maxSIze: number;
                maxAge: string;
                timeout?: string;
                strategy?: 'freshness' | 'performance' //Here
i will keep 'performance' as we are focusing on performance
while adding service workers.
          }
```

- This will help in improving the performance by allowing service workers to intercept all outgoing HTTP requests made by the application and respond to them by following the protocols defined in the ngsw-config.json file.

Refactored

- Currently app-module has all the imports including all the components and provides all the services. This should be restructured using the feature module.
- Under this feature, all the independent components will be grouped together in a separate module. As for example in Auth-module as shown below:

```
@NgModule({
     declarations: [
          LoginComponent,
          SignupComponent,
```

```
    ResetPasswordConfirmComponent,
        ResetPasswordComponent,
        VerifyEmailComponent,
        AuthComponent
    ],
    imports: [
        CommonModule,
        RouterModule,
        AuthRoutingModule,
    FormsModule,
        NavModule
    ],
    exports: [
    LoginComponent,
        SignupComponent,
    ResetPasswordConfirmComponent,
        ResetPasswordComponent,
    VerifyEmailComponent,
        AuthComponent,
        NavModule
    ],
    providers: [
        AuthService
    ]
})
export class AuthModule { }
```

- After that, this feature module will be imported in the main app-module under @NgModule.imports .
- In the same way, the app-routing-module, which contains all the routes can be splitted up into different sub-routing modules. As for example in auth-routing module as shown below:

```
const routes: Routes = [
 {
   path: 'auth',
   component: AuthComponent,
   children: [
     {path: '', redirectTo: 'login', pathMatch: 'full'},
     {path: 'login', component: LoginComponent},
     {path: 'reset-password', component:
ResetPasswordComponent},
     {path: 'reset-password/confirm/:user_id/:reset_token',
component: ResetPasswordConfirmComponent},
     {path: 'signup', component: SignupComponent},
     {path: 'verify-email/:token', component:
VerifyEmailComponent},
     {path: '**', redirectTo: 'login'}
   ]
 },
];
@NgModule({
     imports: [ RouterModule.forChild(routes)],
     exports: [RouterModule]
})
export class AuthRoutingModule {}
```

- ❏ In the similar way, other independent components can be grouped together and the app-module and app-routing-module will get modularised.
- ❏ For getting more information regarding this idea please see #299 EvalAI-ngx.

❏ **Lazy loading for further improvement in the performance.**

- ❏ Just like the service workers, lazy loading is also one of the important features that can enhance the performance of the website to a considerable amount.
- ❏ After separating the main app-module and app-routing-module into smaller sub-modules, I'll implement lazy loading by using "loadChildren" property while defining it's main route.
- ❏ This will make the routes to be called whenever needed(i.e lazily). Currently, all the routes get loaded with the loading of the site which affects the performance of the site.
- ❏ For getting full idea regarding how will i implement it and how it will increase the performance, please refer #303 EvalAI-ngx.

❏ Test covered

- ❏ Currently, the test coverage is around 50%. I will keep on adding test cases along with enhancements and changes to take the coverage at least up to 90%.
- ❏ I have already started writing the missing tests which you can find at #295 EvalAI-ngx. Currently, it increases the coverage by 15%, I will try to raise it up to 40% so that the total coverage remains in the range of 90%.
- ❏ I will also keep on writing missing documentation for the methods.
- ❏ Along with adding proper comments in the code to make it more cleaner and organised.

❏ Robust and Modular

- ❏ Under this feature, I will define custom directives and pipes for the piece of code which are used often.
- ❏ I will write a custom directive for checking whether use is authenticated or not and will replace all the checks with

the directive at all the If condition for checking the user authentication. The custom directive for auth checking will be defined as shown below:

```
@Directive({
 selector: '[appAuthcheck]'
})
export class AuthcheckDirective implements OnInit {

 constructor(
   private authService: AuthService,
   private templateRef: TemplateRef<any>,
   private viewContainer: ViewContainerRef
 ) { }

 condition: boolean;

 @Input() set appAuthcheck(condition: boolean) {
   this.condition = condition;
 }

 ngOnInit() {
   const authVal = this.authService.isAuth;
   if (authVal && this.condition || !authVal &&
!this.condition) {
     this.viewContainer.createEmbeddedView(this.templateRef);
   } else {
     this.viewContainer.clear();
   }
 }
}
```

- ❏ For more detailed information regarding this directive, please see #302 EvalAI-ngx.
- ❏ I will write similar directives for formErrorCheck , canShowPassword and other If conditions in order to

make the code more organized by keeping the logic in one place.

Project Schedule

Time Period	Task Description
Post-proposal period (March 31 - May 04)	➔ Familiarizing to work with Angular 7 and materialize css along with other modules like "@angular/pwa", zipfile etc. that will be required for finishing this project. ➔ Trying out other approaches like lazy loading and service workers to be used in the project to help speed up the work during summers. ➔ Brainstorming on additional endpoints and features that can enhance the UI and functionality of the website. ➔ Solving the issues found while working on the above mentioned features.
Community Bonding (May 04 - June 01)	➔ Get to know the community and the mentors. ➔ Discuss any modifications on existing features. ➔ Discuss new features, and documentation ideas. ➔ I have my end semester exams from May 6 - May 26, which would keep me engaged. But before that, I will be actively available for bonding and discussions.
Week 1-2 (June 01 - June 15)	➔ Starting off with adding the feature to participate in the challenge based on an invitation by the challenge host.

	➔ Starting with the addition of the “Invitation to participate” section in the “challenge settings”. ➔ Writing the function to send an invitation to the particular participant. ➔ Writing test for the component. ➔ Show the invitation to the participant. ➔ Adding the 400, 500 and 404 error pages on the website.
Week 3-4 **(June 15 - June 29)**	➔ Adding a section on the home page for testimonials and statistics. ➔ Writing the function to fetch the feedback from the database. ➔ Writing tests for the component. ➔ Make the required changes for enhancing the UI for displaying the submissions. ➔ Taking mentors’ review and suggestions before the first evaluation ➔ Adding other required improvements as per the mentor’s review.
First Evaluation **(June 29 - July 03)**	➔ Participate based on the challenge, testimonials and statistics and error pages for 404, 400 and 500added. ➔ Tests written for all the components and tested.
Week 5-7 **(July 03 - July 27)**	➔ Making changes to the Previously added features based on the first evaluation. ➔ Starting with the prettier URLs for the challenge. ➔ Adding support for editing challenge details. ➔ CRUD feature to challenge hosts on submissions. ➔ Writing tests wherever necessary while adding these features along with proper testing.

Second Evaluation (July 27 - July 31)	➔ Prettier URLs for the challenge ➔ Adding support for editing challenge details ➔ Adding CRUD features to challenge hosts on submissions. ➔ Well tested unit tests written for the required components.
Week 8-9 (July 31 - August 14)	➔ Making changes to the features added based on the second evaluation. ➔ Adding the feature of filtering the challenges on UI ➔ Exporting submissions in different formats for analysis. ➔ Enhancing the performance of the website by adding service workers. ➔ Adding the required tests where ever required. ➔ Refactoring the code to make it well structured and cleaner. ➔ Adding Lazy loading for speeding up the performance of the site. ➔ Adding pipes and directives as per the requirement.
Week 10 (August 14 - August 24)	➔ A buffer week, in case something does not go as planned. ➔ If everything goes as planned, then this week can be used to fix bugs (if any) or make enhancements to the tests and features added till now. ➔ It can also be used further to enhance the UI and add other minor missing features(if any).
Code Submission and Final Evaluation (August 24 - August 31)	➔ End product ➔ All the required features and improvements done and is also open to be enhanced in future if required. ➔ Well-documented

	➔ Test Covered
Mentors Submit Final Evaluation (August 31 - September 07)	➔ Mentors submit their final evaluations

Time

My typical working hours are 11:30-13:30, 15:00-19:00, 22:00-03:00 (IST) i.e 6:00-8:00, 9:30-13:30, 16:30-21:00(UTC) and on an average, I will be able to give minimum 40-50 hours per week on this project.

Any Other commitments during the coding period time?

I don't have any other commitments in the summer and I will be staying back home for the most part of it. So, I would be able to give all my time for this project only for the complete 3 months.

Why am I the right person for this project?

- ❏ I have been working with Angular 7 for 3 years from now, which is the main skill requirement for this project. Along with this, I am also comfortable with Html, materialize css and Typescript which would be the basic requirement for the project.
- ❏ I am continuously contributing to the EvalAI-ngx and EvalAI since January. So, I am very well acquainted with its code base and 'll be able to work faster on this idea as I am very well aware of the features introduced by the particular part of its code base.
- ❏ I have worked with many Angular based websites as mentioned in the Experiences section above.

- ❏ I am working and have already worked on most of the features introduced in this idea. I have made WIP(Work In Progress) PR for some of the features I implemented. You can take a look at these PRs here. I have also attached the code snippets for the other features of this project based on the research I have done.
- ❏ I have already worked on each of the ideas and have decided the path of how to proceed by doing all the necessary searching, implementing, understanding EvalAI and EvalAI-ngx codebases and referring docs, it will help me to work faster on this project during the coding period.
- ❏ I have chosen this project for contribution due to my interest and so will continue to work on it as a part of my learning process. Also being a fast learner, I will learn about the required modules and packages faster and try to get most out of it.

My plan after GSOC

- ❏ To shift the code base to the latest version of Angular i.e from Angular 7 to Angular 9. As the Angular 9 is out with many features like the default IVY compiler which reduces the bundle size over 25-40 %. It also checks bindings and report issues, which helps in detection of bugs in the development environment. So, shifting to Angular 9 will make the site more light and will also enhance its speed.
- ❏ To further enhance the user experience by improving the UI and adding other missing small features like adding “leaderboard precision value”, missing feature in challenge host setting.
- ❏ To continuously work on improving the functionalities of the site as per the feedback received from the participant and challenge hosts.
- ❏ To fix minor left out issues like “Edit host team should show the pre-filled data” and continuously work on finding and solving the issue to keep the website bug free and up to date.

nithyanithin210@gmail.com

References:

I have referred the following links while working on this idea:

- https://angular.io/guide/service-worker-getting-started
- https://www.npmjs.com/package/@angular/pwa
- https://blog.angular-university.io/service-workers/
- https://angular.io/guide/structural-directives
- https://malcoded.com/posts/angular-fundamentals-modules/
- https://angular.io/guide/feature-modules
- https://nitayneeman.com/posts/restructuring-an-angular-application-with-feature-modules/
- https://pusher.com/tutorials/lazy-loading-angular-7
- https://thinkster.io/tutorials/building-real-world-angular-2-apps/authentication-based-directives
- https://angular.io/guide/pipes
- https://codecraft.tv/courses/angular/pipes/custom-pipes/
- https://angular.io/guide/testing
- https://medium.com/swlh/angular-unit-testing-jasmine-karma-step-by-step-e3376d110ab4
- https://docs.python.org/3/library/zipfile.html

nithyanithin210@gmail.com