**Topcoder - Elasticsearch Feeder Service For Challenges version 1.0**

**Verification Guide**

# 

# Setup

For the challenge feeder service, run 'mvn clean package' to compile the code and then cd local folder and run './run.sh' to start the services. Before starting the services, you need to first setup the envrionment as following sections.

If you want to push the challenge to the amazon elasticsearch service, configure the

the local/run.sh as:

#export ELASTIC\_SEARCH\_URL="http://$DOCKER\_IP:9200"

export ELASTIC\_SEARCH\_URL="https://search-your-url-xxxxx.us-east-1.es.amazonaws.com"

export CHALLENGES\_INDEX\_NAME="challenges"

export AWS\_SIGNING\_ENABLED=false

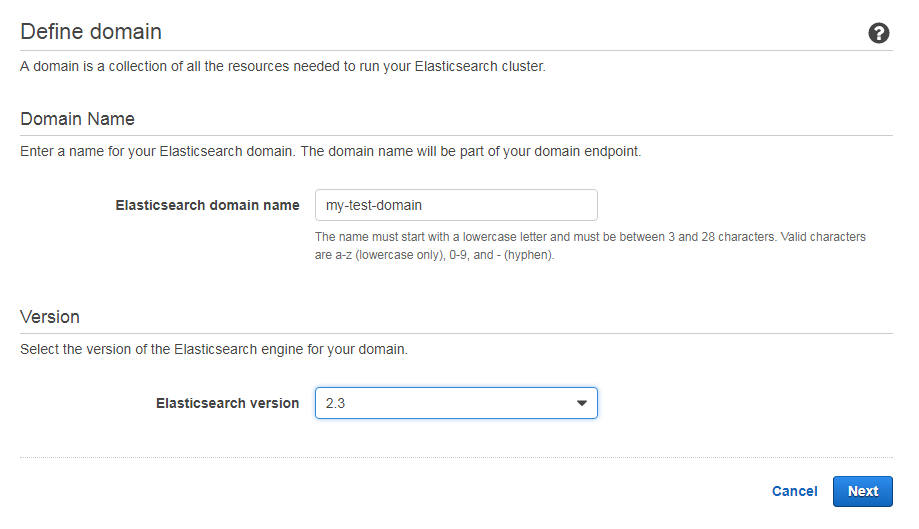
export AWS\_REGION=us-east-1

export ASW\_SERVICE=es

export AWS\_ACCESS\_KEY=your key here

export AWS\_SECRET\_KEY=your secret here

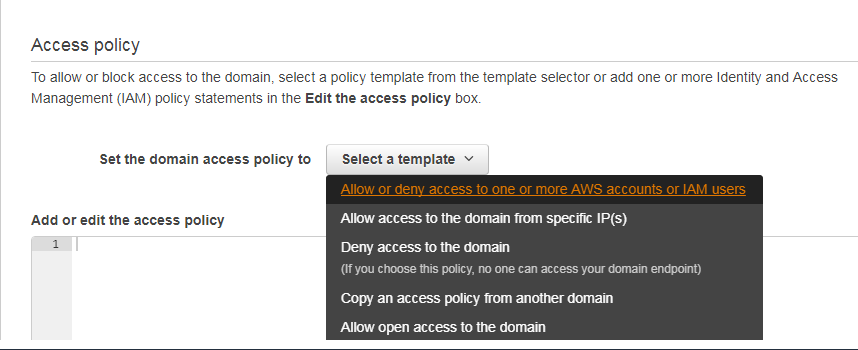
You should create the domain at https://aws.amazon.com/cn/elasticsearch-service/.



When you are setting up access policy there are many options.

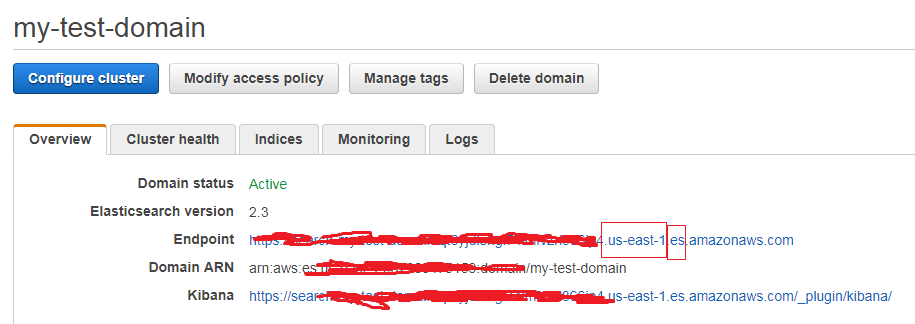
For example if you choose **Allow or deny access to one or more AWS or IAM users**.

You must configure **AWS\_ACCESS\_KEY**, **AWS\_SECRET\_KEY**, **AWS\_SERVICE**, and  **AWS\_REGION** in local/run.sh, and set **AWS\_SIGNING\_ENABLED** as true.



If you chose 'Allow open access to the domain', set AWS\_SIGNING\_ENABLED to false to disable the credential. Please been noted that this should be just for test purpose.

After you wait about 10 minutes to create domain successfully you can configure **ELASTIC\_SEARCH\_URL** to the endpoint below:



The **AWS\_REGION** is us-east-1 and the **AWS\_SERVICE** is es.

The direct app docker is https://github.com/appirio-tech/tc-common-tutorials/tree/master/docker/direct-app

Please follow the above url to configure and setup the tc direct app docker and online review docker.

For the elasticsearch, clone this branch: https://github.com/topcoder-platform/challenges-logstash-conf/tree/populate\_more\_challenge\_info, configure and setup as Populate-Challenge-Submissions-in-ElasticSearch\_verification.docx. Every time you change the data in the database, it's necessary to run one of the following two commands to update the elastic services:

docker-compose up challenges-feeder-initial-load

docker-compose up challenges-feeder

Please refer to the Populate-Challenge-Submissions-in-ElasticSearch\_verification.docx for details

Then execute the following sql for the tcs\_catalog database to make the users agree to all terms:

INSERT INTO user\_terms\_of\_use\_xref

SELECT u.user\_id, t.terms\_of\_use\_id, CURRENT, CURRENT

FROM user u

JOIN terms\_of\_use t ON 1=1

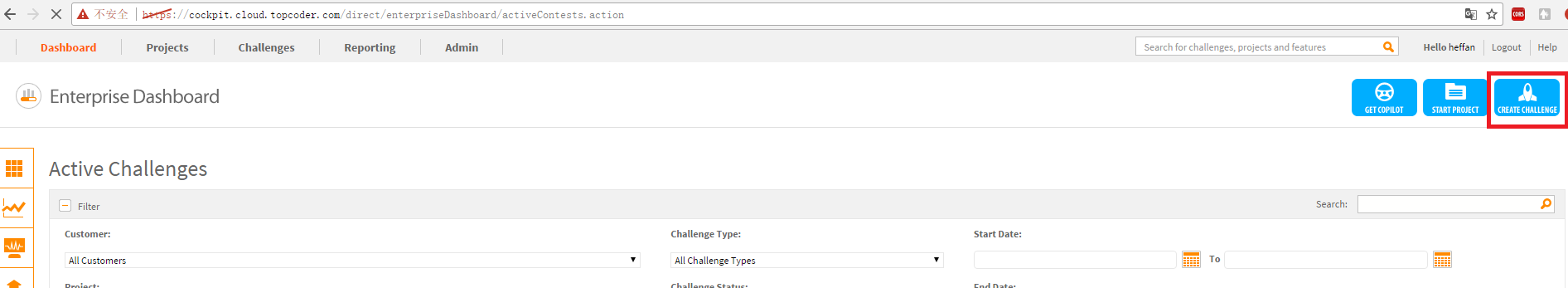
LEFT JOIN (SELECT user\_id, terms\_of\_use\_id FROM user\_terms\_of\_use\_xref) x

ON x.user\_id = u.user\_id AND x.terms\_of\_use\_id = t.terms\_of\_use\_id

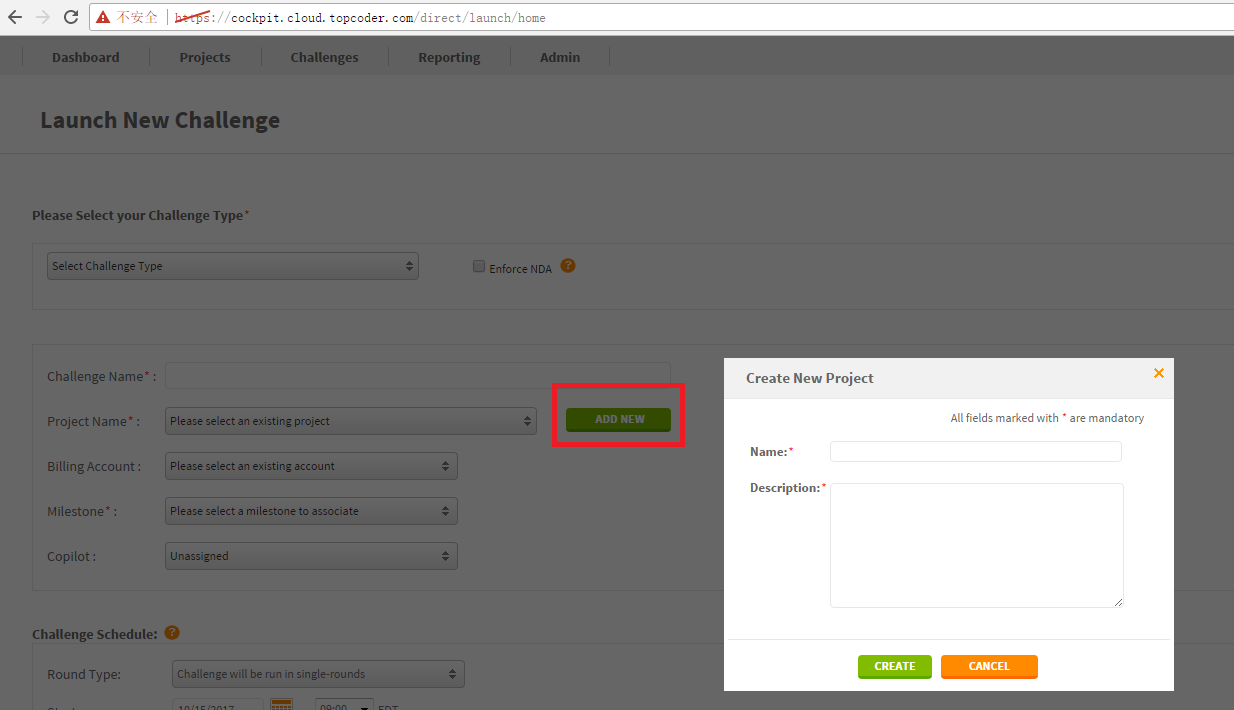
WHERE x.user\_id IS NULL;

The following sections just introduce the workflow of how to create a dev type challenge(assembly challenge) and move it to the final fix phase with the online-review system, the phase time for the challenge is not updated.

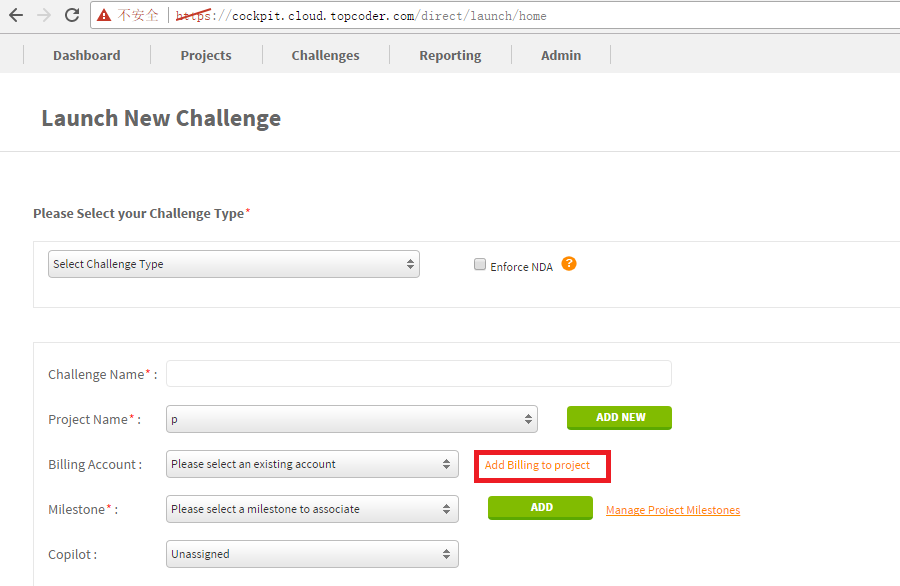
Go to https://cockpit.cloud.topcoder.com/direct and login as heffan/password, then click the Create Challenge



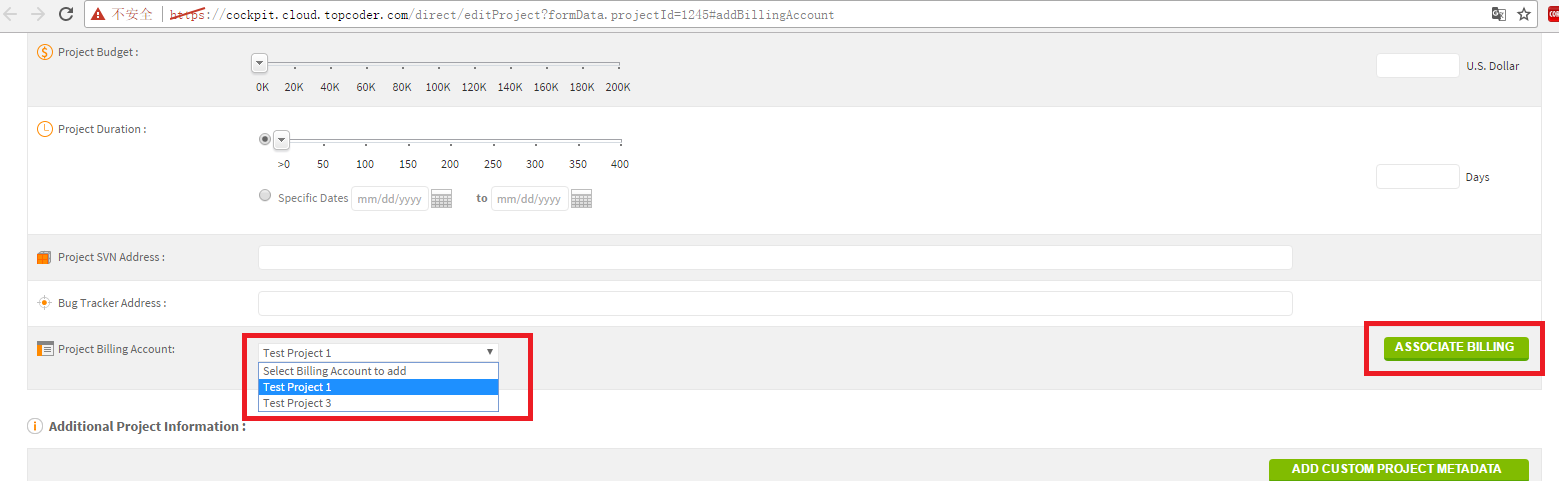
Then click the ADD NEW to create a new project.



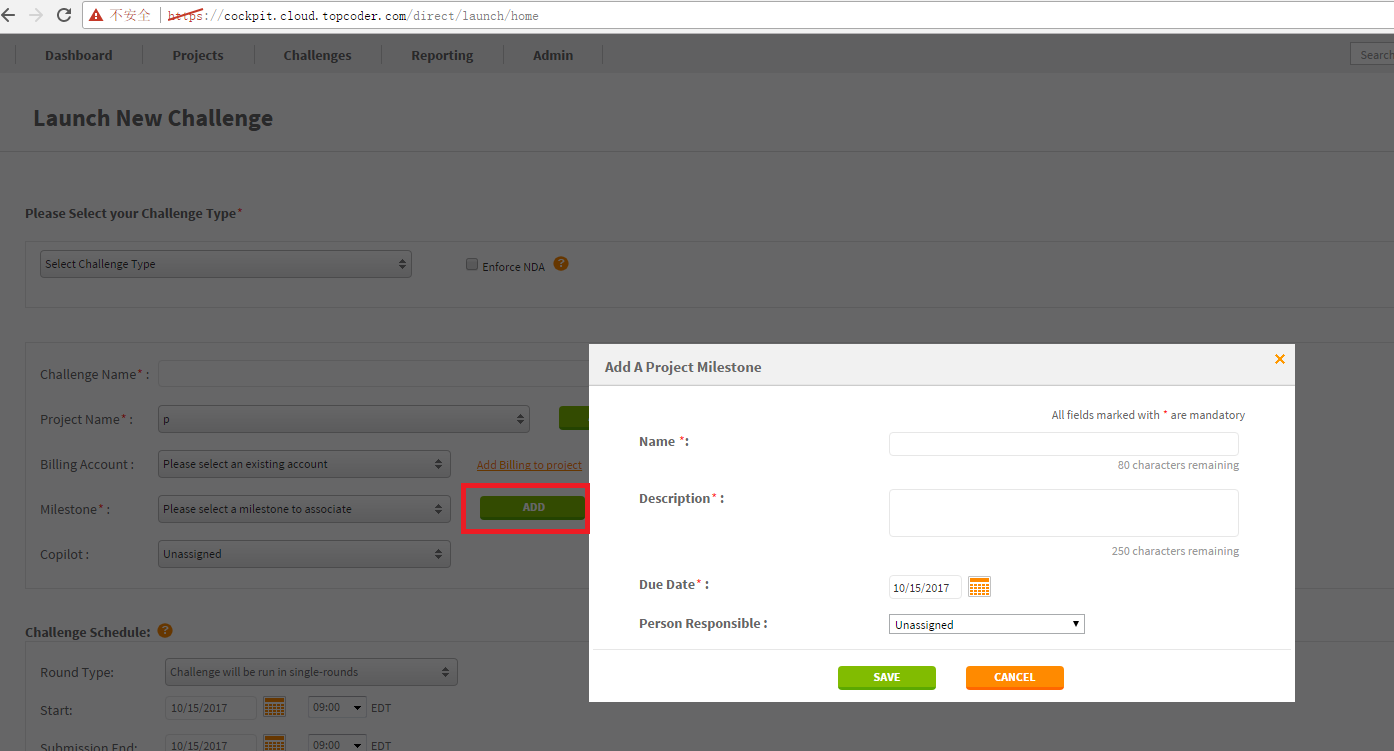
Click the Add Billing to project to add the billing:



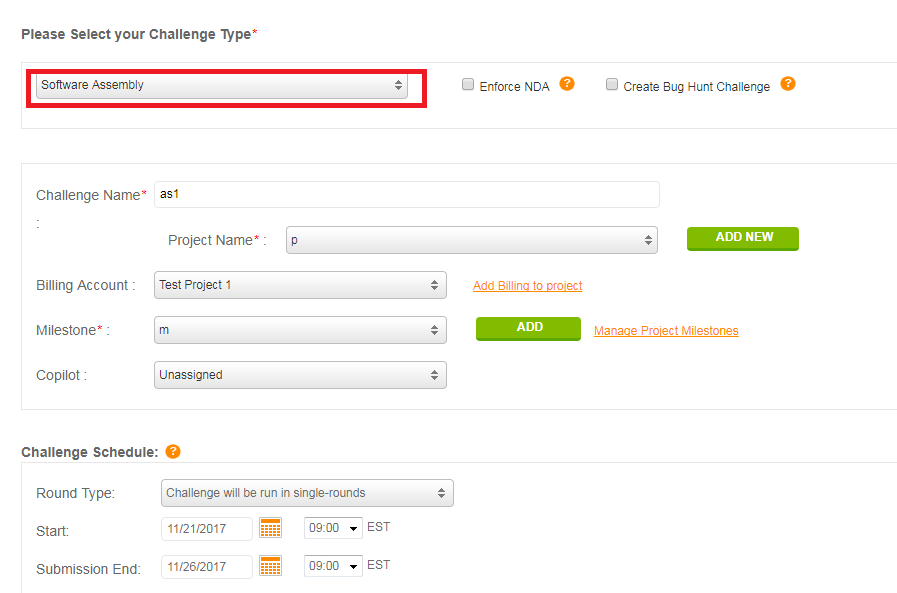
In the new page, select the project and click the ASSOCIATE BILLING BUTTON:



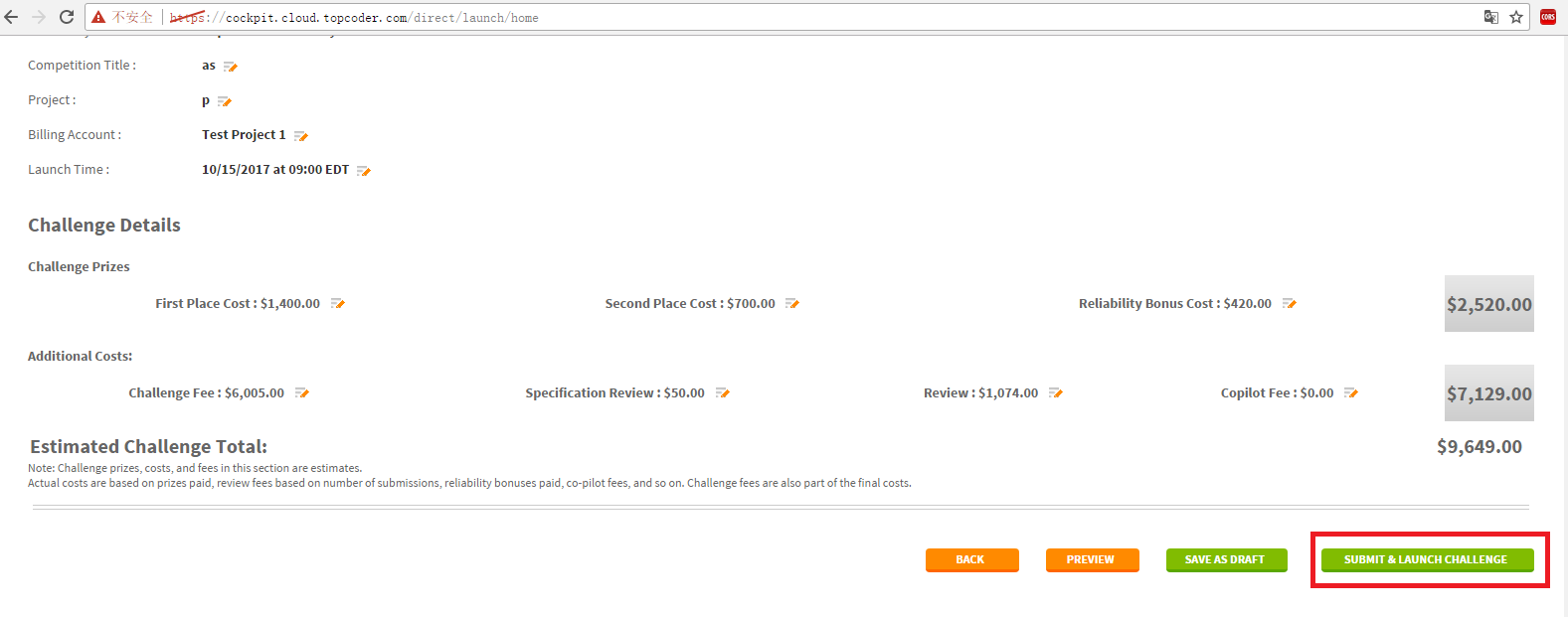
Then refresh the https://cockpit.cloud.topcoder.com/direct/launch/home again and add a milestone:



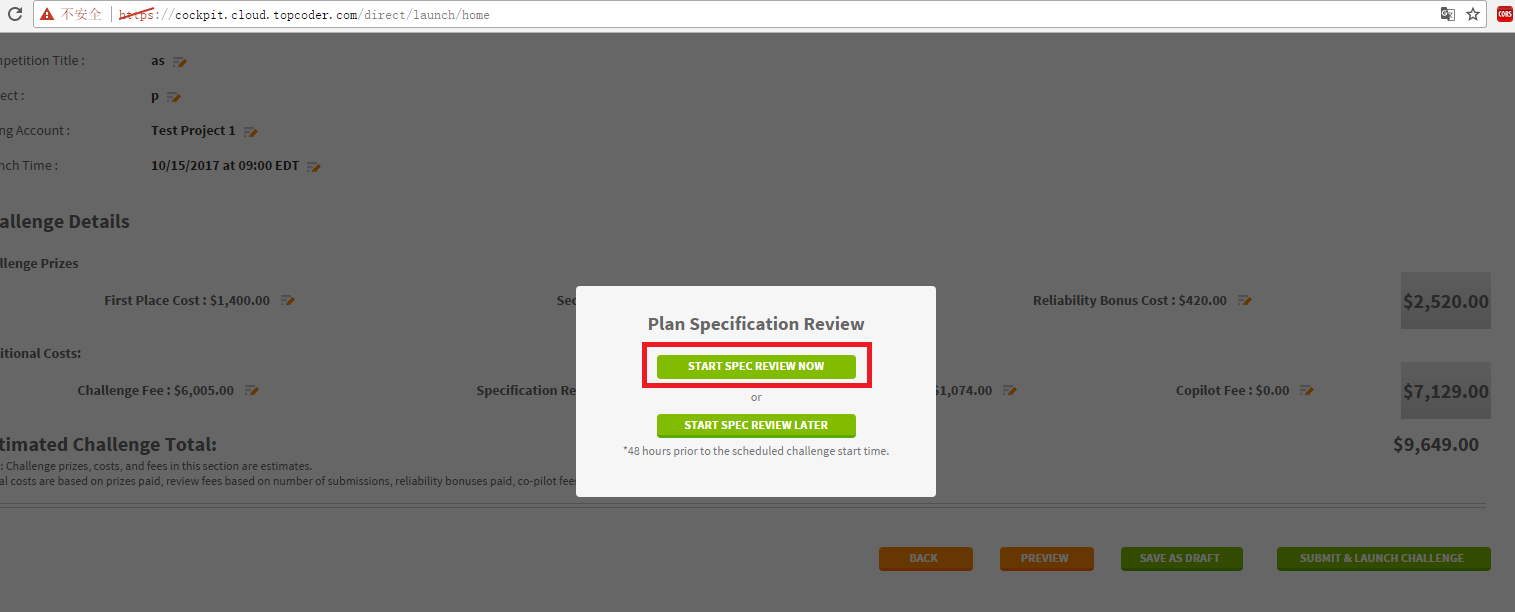
Then create an assembly challenge and fill any required data and click the continue button, please select the multi-rounds if you want the checkpoints phase.



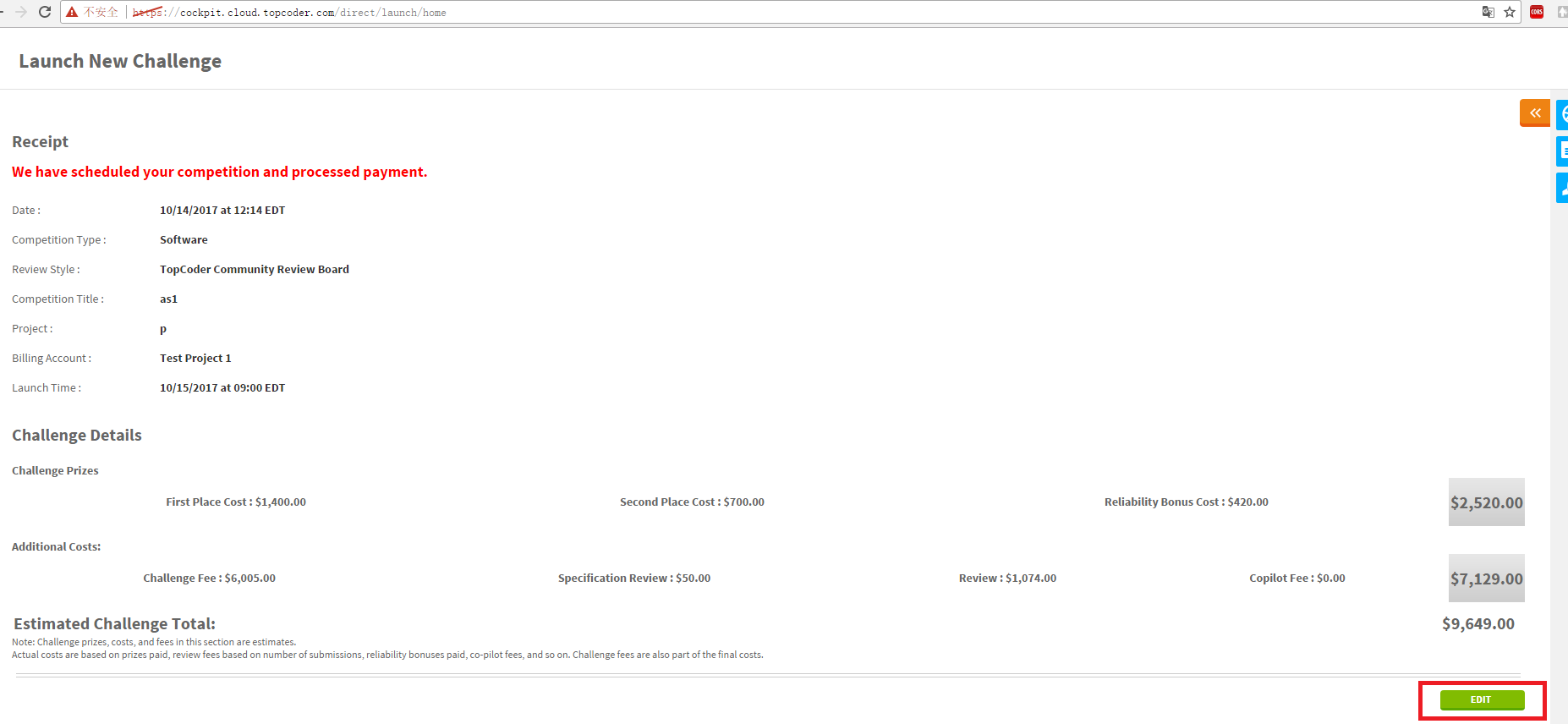
Finally click the submit button below:



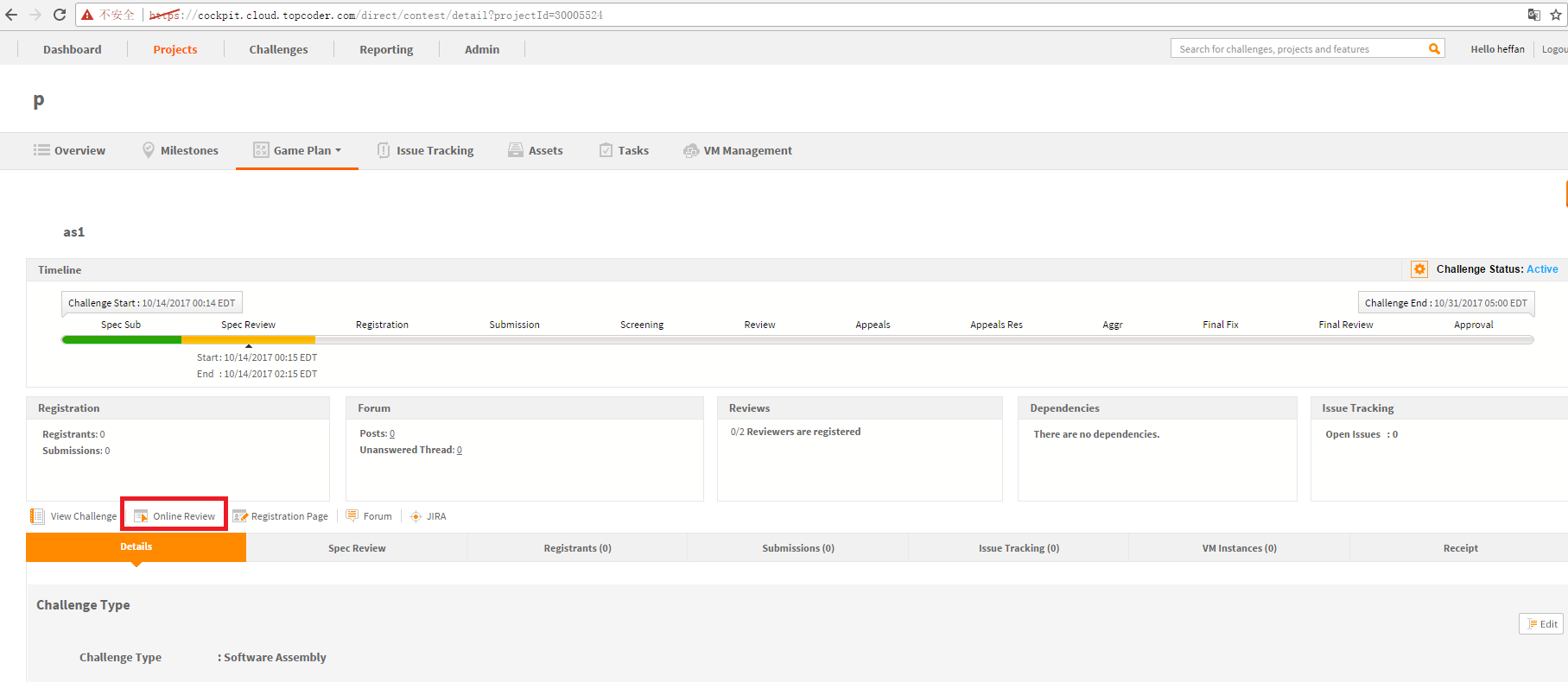
Click start the spec review now:



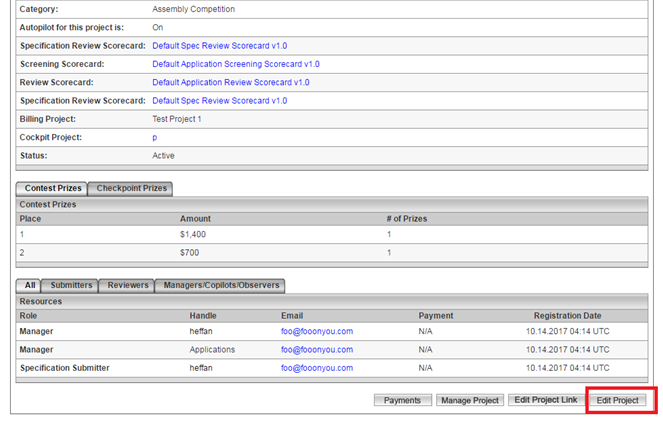
Then click the edit button:



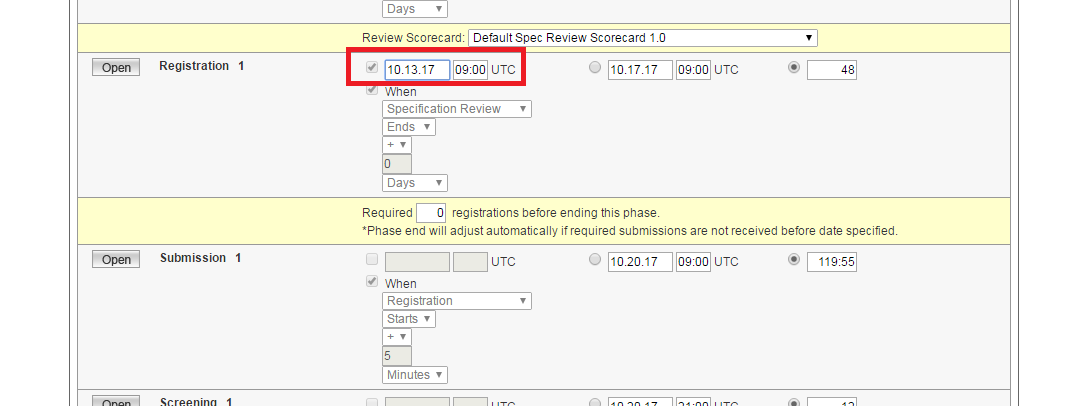
Then click the Online Review to go to the online review system



The url for the online review should be like http://tcs.cloud.topcoder.com/review/actions/ViewProjectDetails?pid=30005520

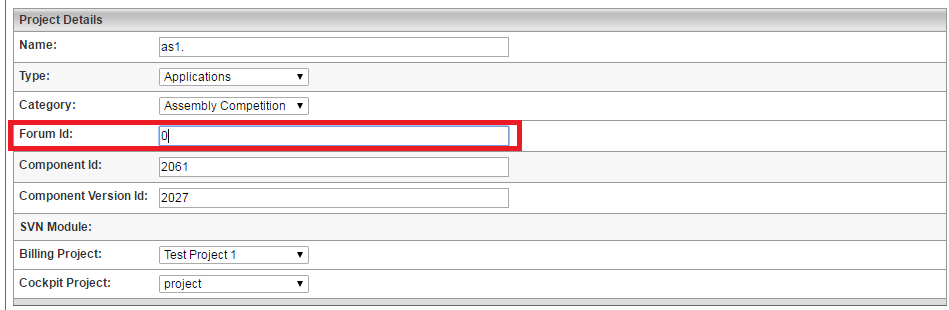


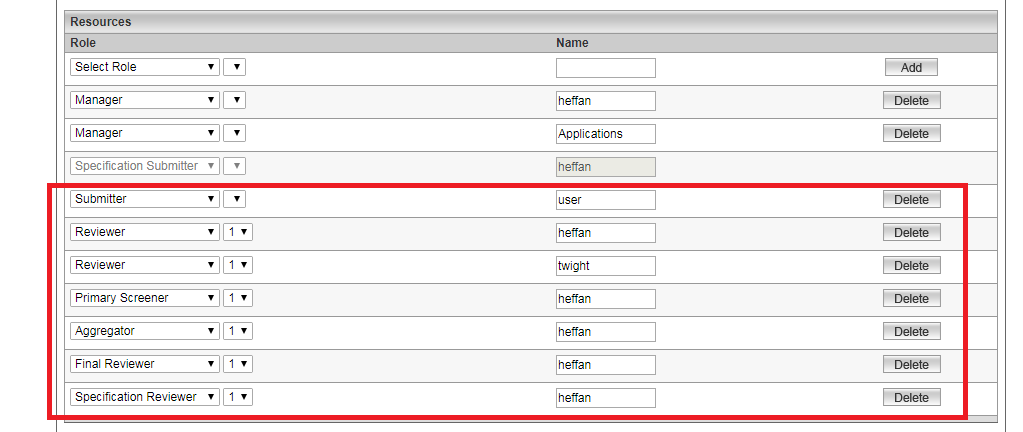
Then click the Edit Project button:



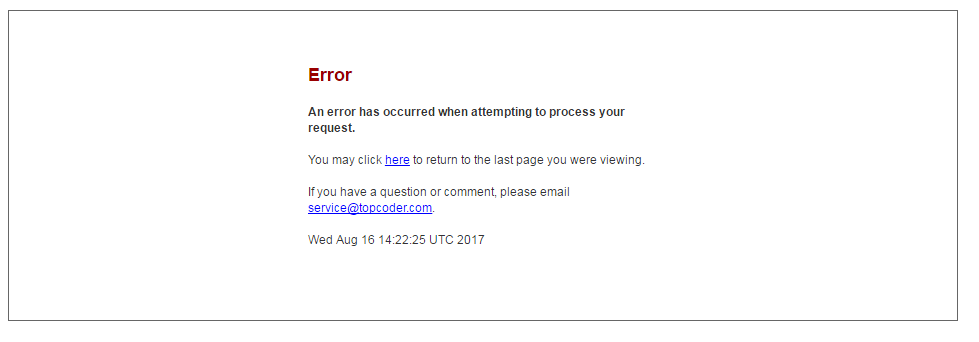
Edit the registration start date as above to open the registration phase(make the start date before the current date)

Then scroll down, add the forum id, and the spec reviewer, submitter and reviewers as following:



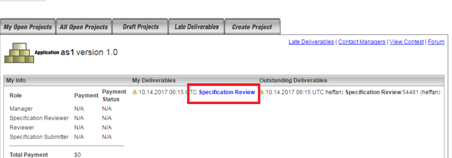


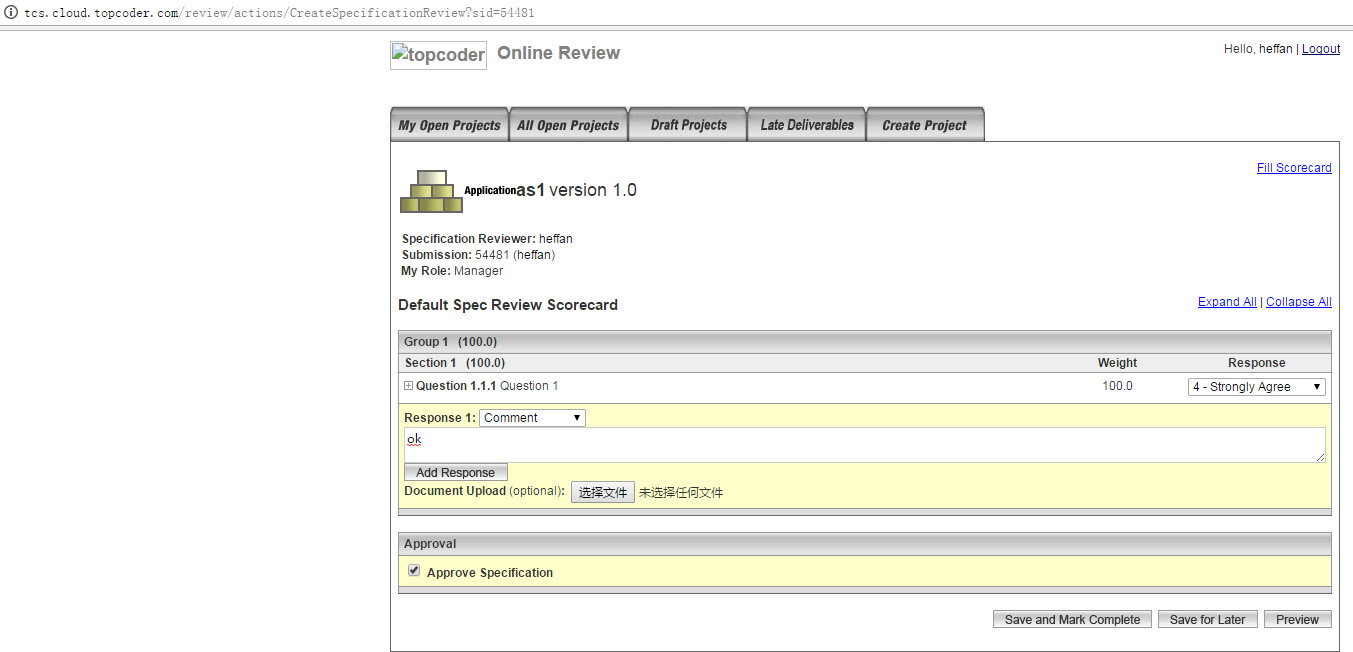
Then click the save changes button below, ignore the errors after save:



Then refresh the http://tcs.cloud.topcoder.com/review/actions/ViewProjectDetails?pid=30005520, there should be 'Specification Review' link.

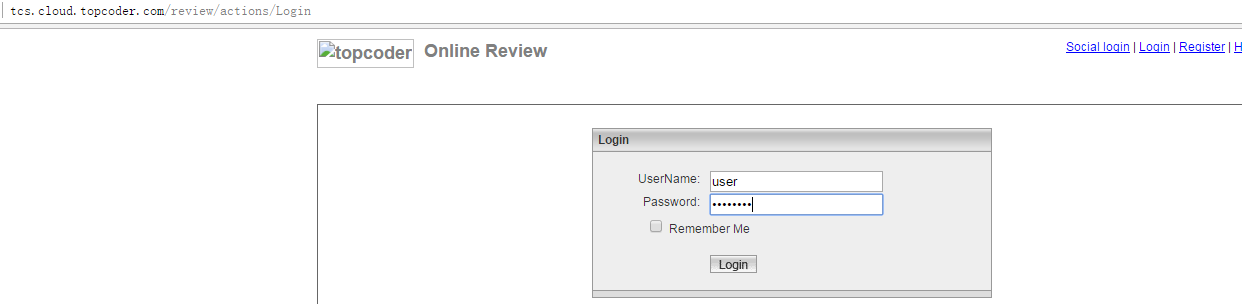
Let heffan finish the spec review:



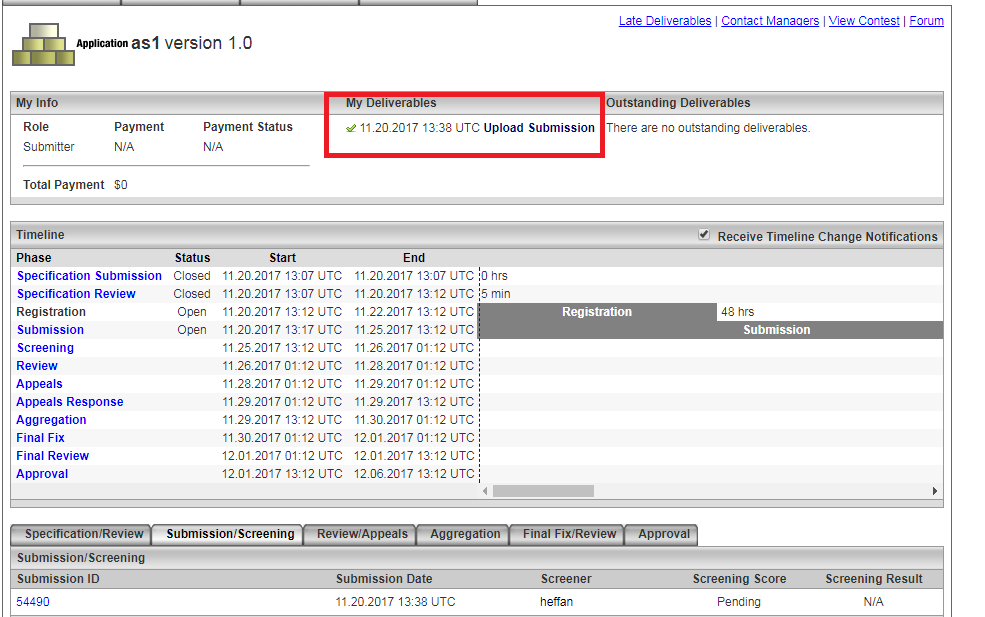


Then refresh the page http://tcs.cloud.topcoder.com/review/actions/ViewProjectDetails?pid=30005520 again, the registration phase should be started. Wait for about 5 minutes for the submission phase starts.

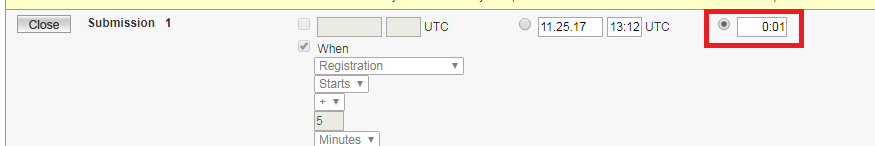
Then click the Logout button above and login as user/password



Then go to http://tcs.cloud.topcoder.com/review/actions/ViewProjectDetails?pid=30005520, and upload the submission:



After user upload the submission, login as heffan, and go to the edit project page and set the submission duration phase to 0:01 to end the submission phase:



Let heffan finish the screening and review scorecards.

Login as twight and finish the review scorecards

Login as user and do some appeals and then complete the appeals.

Login as heffan to finish the appeal response.

Then move the project to aggregation phase and final fix phase.

You can also run the following sql for tc\_catalog database to move the phase to final fix if the response phase is late or pending:

update project\_phase set phase\_status\_id = 2 where project\_id=30005520 and phase\_type\_id = 9;

update project\_phase set phase\_status\_id = 3 where project\_id=30005520 and phase\_type\_id < 9;

--Then login as user and let user upload final fix.

--Then attach the events:

insert into tcs\_catalog:contest (contest\_id, contest\_name, event\_id, modify\_date) values (111111, 'test', 993, current);

insert into tcs\_catalog:contest\_project\_xref (contest\_id, project\_id, create\_date) values(111111, 30005520, current);

update tcs\_catalog:project\_info set value = 1000 where project\_id = 30005520 and project\_info\_type\_id = 30;

--Then set the groups:

delete from group\_contest\_eligibility where contest\_eligibility\_id in (1, 2);

delete from contest\_eligibility where contest\_eligibility\_id in(1,2);

insert into contest\_eligibility values (1, 30005520, 0);

insert into contest\_eligibility values (2, 30005520, 0);

insert into group\_contest\_eligibility values(1, 10);

insert into group\_contest\_eligibility values(2, 14);

update security\_groups set challenge\_group\_ind = 10 where group\_id = 10;

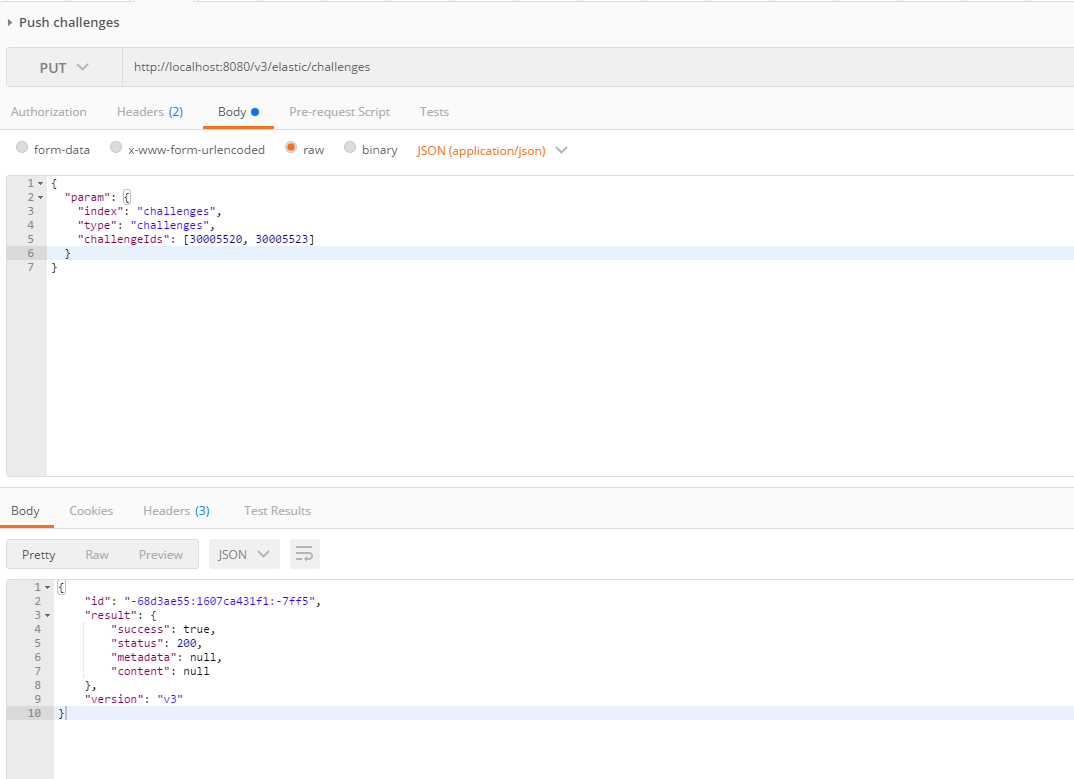
update security\_groups set challenge\_group\_ind = 14 where group\_id = 14;

# Verify

Import the docs/ challenge-feeder.postman\_collection.json

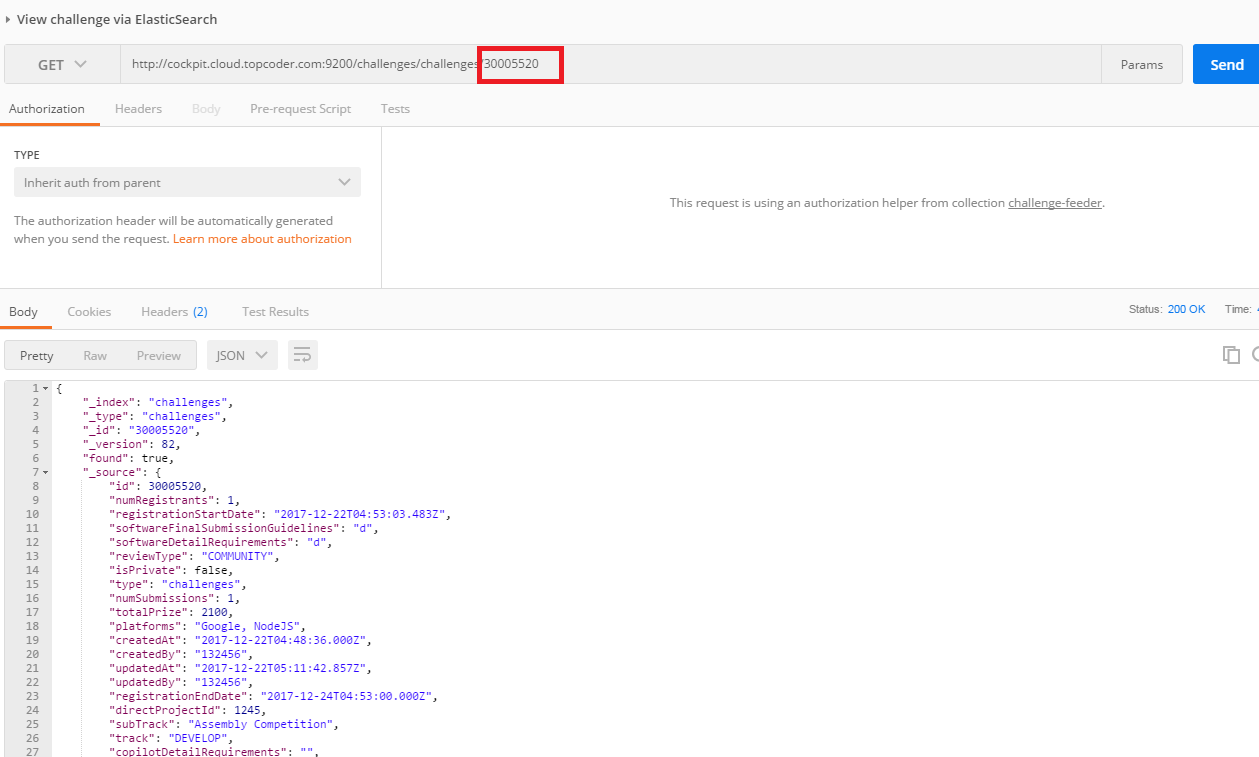
Push challenges

The id 30005520 is for assembly in the final fix. 30005523 is a code challenge in the registration phase.

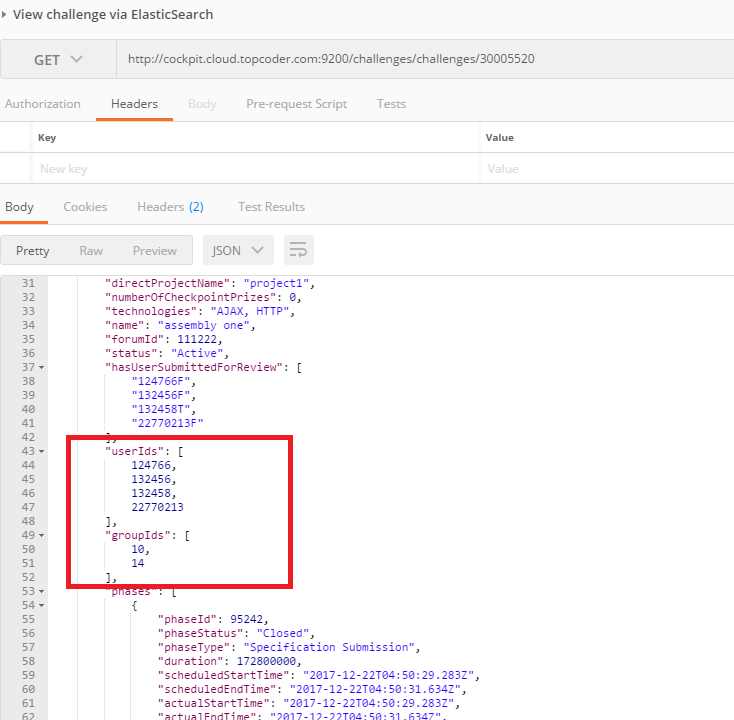


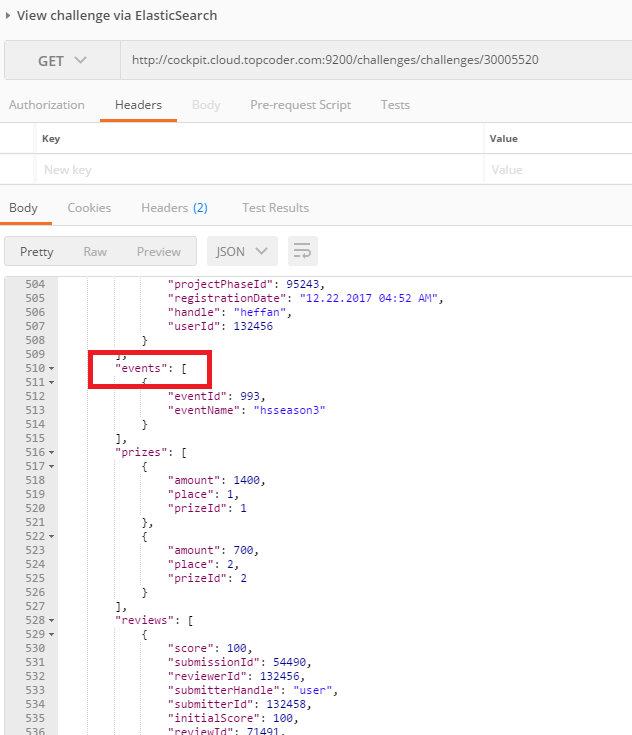
View challenge via ElasticSearch

Change the challenge id in the red rectangle.



scroll down to check other fields:

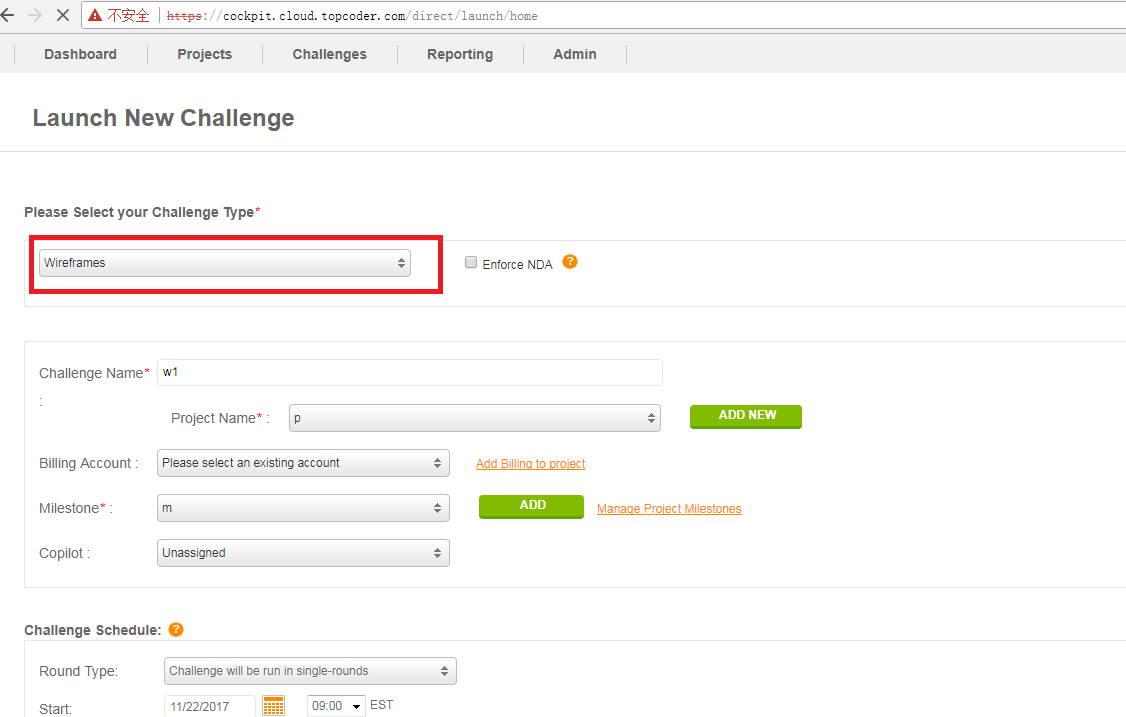






For the code challenge, check it in the same way. If you configure as push the data to the amazon service, change the http://cockpit.cloud.topcoder.com:9200 to https://search-your-url-xxxxx.us-east-1.es.amazonaws.com

For the design challenge, go to direct app and create a wireframe challenge:



And fill the other fields and launch the challenge, assume that the challenge id is 30005522

In the online review, we can not upload the submission for the design(or studio) challenge, the upload submission link will be redirected to the url: http://studio.cloud.topcoder.com/?module=ViewContestDetails&ct=30005522

Hence we use the sql to add the submissions for the user, you just need to replace the 30005522 with the id you created below.

Select the tcs\_catalog database and run the following sql.

--Run the sql to open the submission phase:

update project\_phase set phase\_status\_id = 3 where project\_id = 30005522 and phase\_type\_id in (1, 13, 14, 15, 16, 17);

update project\_phase set phase\_status\_id = 2 where project\_id = 30005522 and phase\_type\_id = 2;

--Run the sql to add the user as the submitter

insert into resource values(99026, 1, 30005522 , null, 132458, 132456, current, 132456, current);

insert into resource\_info values(99026, 2, 'user', 132456, current, 132456, current);

insert into resource\_info values(99026, 1, 132458, 132456, current, 132456, current);

--Run the sql to upload the submission for user:

insert into upload values(99031, 30005522 , (select project\_phase\_id from project\_phase where project\_id=30005522 and phase\_type\_id=2), 99026, 1, 1, 'xxx', null, 132458, current, 132458, current);

insert into submission values(99031, 99031, 1, null, null, null, null, 1, 132458, current, 132458, current, 1, null, null, null, null, null, null);

--add the heffan as the reviewer

insert into resource values(99225, 4, 30005522 , null, 132456, 132456, current, 132456, current);

insert into resource\_info values(99225, 2, 'user', 132456, current, 132456, current);

insert into resource\_info values(99225, 1, 132456, 132456, current, 132456, current);

-- move the project

update project\_phase set phase\_status\_id = 2 where project\_id=30005522 and phase\_type\_id = 11;

update project\_phase set phase\_status\_id = 3 where project\_id=30005522 and phase\_type\_id < 11;

--update the submission score

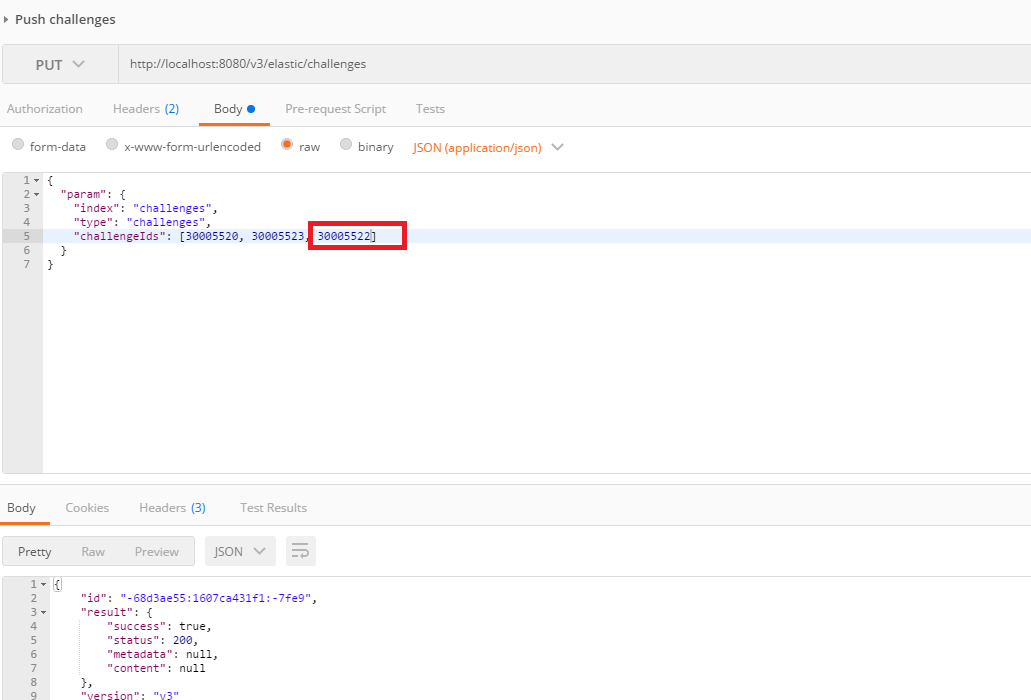
update submission set final\_score = 100, placement=1, screening\_score = 100, initial\_score= 100 where upload\_id in (select upload\_id from upload where project\_id in(30005522 ));

--run the following sql to init review data

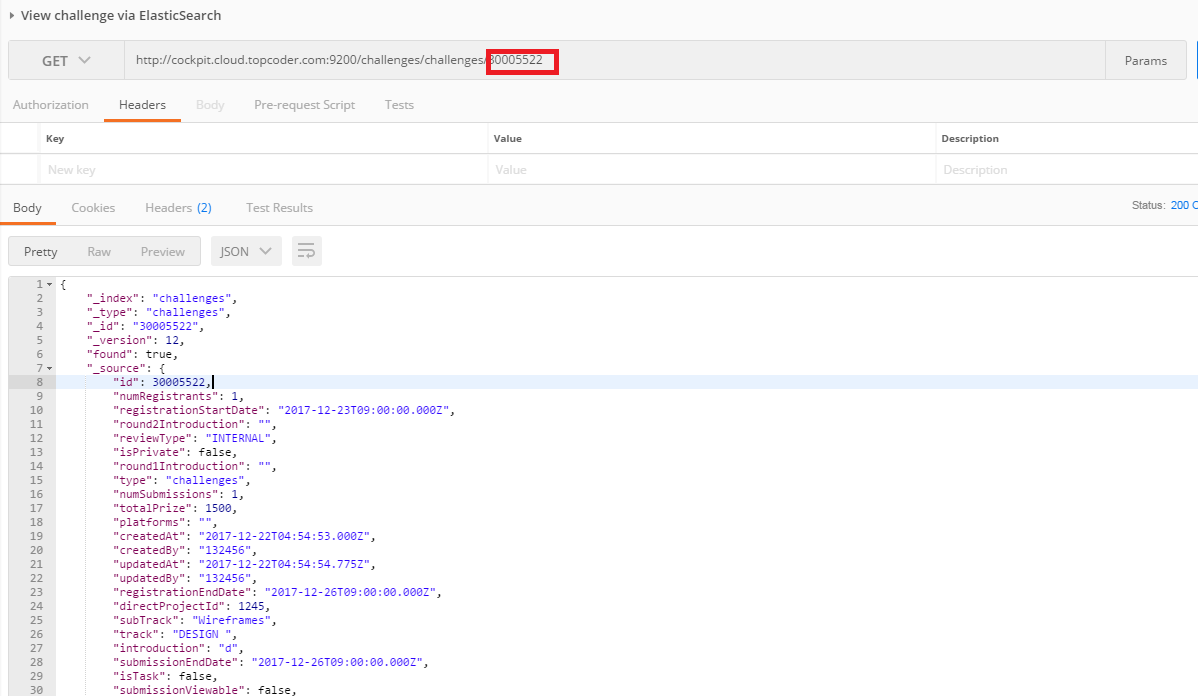
insert into review (review\_id, resource\_id, submission\_id, project\_phase\_id, scorecard\_id, committed, score, initial\_score, create\_user, create\_date, modify\_user, modify\_date)

values (90001, 99225, 99031, (select project\_phase\_id from project\_phase where project\_id = 30005522 and phase\_status\_id=2), 30000410, 1, 100, 100, 132456, current, 132456, current);

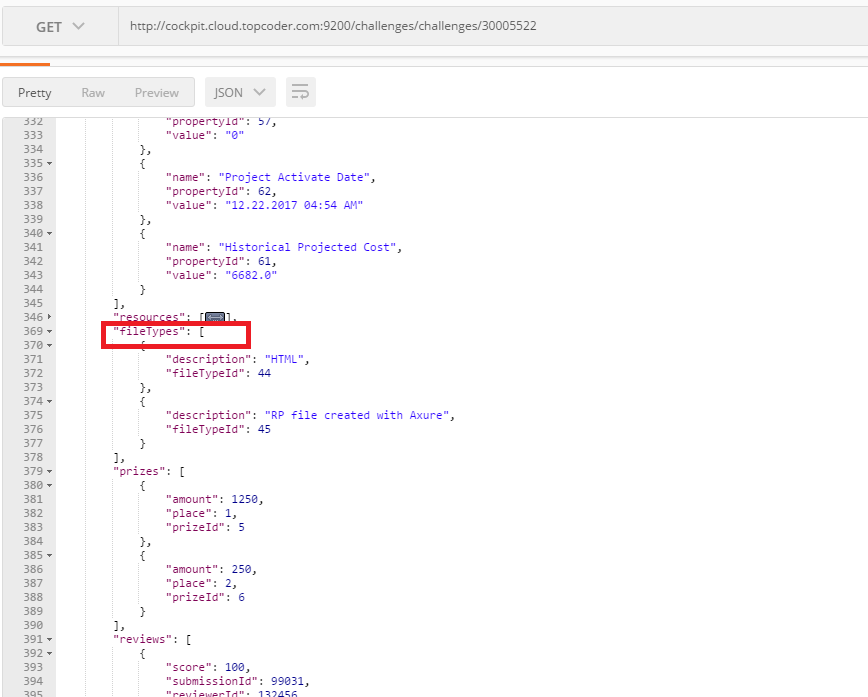
Then push the challenges:



Check it in the elasticservice:



scroll down:



Then, you can run docker-compose up challenges-feeder-initial-load from the cloned challenges-logstash-conf to check that it pushes the same challenge data.

For other invalid request, check them in the same way.