GSoC-2023 Project Proposal
Evaluation Infrastructure Optimization

1 Basic Details

- **Full Name:** Ayush Rathore
- **Location:** Bilaspur, Chattisgarh, India
- **Time Zone:** UTC +5:30 Indian Time Zone
- **Education:** B.Tech in Computer Science & Engineering at Rajiv Gandhi Institute of Petroleum Technology (Batch of 2024)
- **Email:** 20cs3018@rgipt.ac.in
- **GitHub:** ayushr1
- **Skype:** live:.cid.97082b778ed2b0f9
- **Portfolio:** ayushr1.netlify.app
- **Resume:** Ayush-Rathore-Resume.pdf

2 Statement of Motivation

2.1 What is your motivation for participating in Google Summer of Code?

Being part of the Google Summer of Code is an exceptional opportunity to boost my career, get global recognition, gain valuable experience in software development, and contribute to organisations that are solving real-world challenges. The prospect of working alongside industry experts and collaborating with top open-source organisations is genuinely thrilling. As an ardent software developer, I have always been driven to solve real-world problems, and my passion for open source has been my driving force since my high school days when I got my first Android phone and got into the world of open source. My enthusiasm for solving problems, fixing existing bugs, and adding features to open-source software has never wavered. I am motivated to participate in the Google Summer of Code because it presents a remarkable chance to tackle challenging projects that address actual problems faced by society and learn a lot of new things which will help immensely in my career. Under the guidance of experienced mentors, I will learn new skills, work collaboratively with developers from diverse backgrounds, and enhance my communication abilities. Contributing to open-source projects and the Google Summer of Code is the perfect platform to achieve my goals.

2.2 Why did you choose CloudCV?

The initial spark for my interest in CloudCV came from my skills, prior experience, and aligning interests. In particular, I was drawn to the concept and use case of EvalAI, which motivated me to delve into its codebase and gain a better understanding of its inner workings. I discovered EvalAI which aims to build a centralized platform to host, participate, and collaborate in Artificial Intelligence (AI) challenges organized around the globe and hope to help in benchmarking progress in AI, with numerous participants, submissions, and evaluation metrics while providing hosts with the flexibility to customize their challenges as needed. Once I started working on this project, I knew that I couldn't turn back. These experiences provided me with valuable insights into the technical aspects of the project and made me capable to contribute to its growth and development through the Google Summer of Code program.

2.3 Why this project idea?

I have chosen Evaluation Infrastructure Optimization as my project idea. I am interested in this project because it aligns perfectly with my past experience and current interests. I have worked on similar projects in the past, including contributing to CloudCV in a similar domain, and I believe that my skills and expertise will be valuable to the team. This project presents an excellent opportunity for me to apply my knowledge and skills on a practical scale in a domain that I am deeply interested in. Setting up automation, integrating with AWS, and building a comprehensive test suite for software is something that I have worked on before, and I find it challenging and rewarding. Overall, I am excited to be part of this project and contribute my skills to ensure its success. I am confident that my experience and enthusiasm make me a suitable candidate for this project, and I am eager to make a valuable contribution to the team.

2.4 What are your expectations from us during and after the successful completion of the program?

I am truly excited to be a part of this community and contribute to the work at CloudCV. First and foremost, I am eager to forge lasting bonds with members of the CloudCV family. Having a welcoming environment that encourages candid communication and teamwork is crucial to me. I think that by collaborating closely with my mentors and other contributors, I may benefit from their knowledge, improve my comprehension, and develop myself.

Second, I'm determined to provide the CloudCV project with useful contributions. I am certain that my knowledge and abilities can benefit the project, and I am willing to put in a lot of effort to make sure I achieve the project's objectives. I appreciate criticism and ideas for my work since I firmly think that both professional and personal development benefits much from constructive critique.

Finally, I pledge to keep lines of communication open with my mentors and the larger community. To make sure that everyone is on the same page and that the project is progressing successfully, I think constant communication is crucial. Regular communication will allow me to ask for help when I need it, advance more quickly, and produce better work.

Finally, I'm excited to go on contributing to the open-source community through

CloudCV after the Google Summer of Code programme is a success. I think the GSoC project is a great chance for me to learn and progress as a developer, and I'm enthusiastic about the idea of taking on bigger tasks and having a real impact on the project's future.

2.5 What are you hoping to learn?

I am excited at the prospect of gaining valuable experience working alongside seasoned professionals on a significant project like CloudCV. Developing my development skills is particularly enticing, and I am eager to build a test suite to familiarise myself with the more complex components of the program and further expand my development knowledge. Moreover, I am confident that staying connected with qualified individuals in the community will provide me with an incredible opportunity to learn and grow. Collaborating on this project has been a great experience, and I have already gained insights into the best practices that we need to follow to maintain high-quality code. Furthermore, I believe that staying connected with such qualified individuals in the community would be a remarkable learning opportunity for me. It has been a great experience collaborating on this project and learning about the practices we need to follow to maintain high-quality code. The codebase is well-managed, with various checks integrated to ensure that syntax and formatting errors are caught before merging a new pull request. These practices have made it easier for new contributors to get started with the project. This experience will help me gain a more comprehensive understanding and take my skills to the next level. I am excited to contribute to this project and make a positive impact.

3 Experiences

3.1 Have you taken Computer Vision, Machine Learning, Artificial Intelligence, Natural Language Processing, and Deep Learning courses?

In my computer science program, I've had the opportunity to study courses in data analysis and statistical methods, data mining, artificial intelligence, and deep learning., I've even completed courses on machine learning from AndrewNG from Coursera (certificates) to deepen my knowledge. I also have elementary exposure to Computer Vision and have made projects utilising MATLAB libraries to build intelligent traffic signal time based on congestion.

3.2 What kind of projects have you worked on in the past? What technologies did you use?

I have experience in development in the backend, frontend, cloud, automation, and Linux kernel development with experience building 10+ full-scale projects and many small projects which are listed below.
I have used technologies such as Python, Django, Flask and JavaScript, and NodeJS for Backend. HTML/CSS, ReactJS for frontend. I have experience setting up infrastructure on AWS for web apps and also have set up CI/CD pipelines. The list of my projects is as follows

Upastithi **- Django, Python, MySQL** - 🔗 </>

- ◦ Composed a Web App performing **CRUD** operations to digitally log attendance of RGIPT students, **deployed** and in use, enabling 2+ faculties, 5+ tutors and **300**+ students to log, monitor & track attendance.

Saarthi **- NodeJS, ReactJS, MongoDB** - 🔗 </>

- ◦ Constructed an education platform ↗ with courses fetched from **YouTube REST API** with a progress tracker and notes sharing community inventory that increased productivity by 10%.

Cpp2LaTeX **- Flask, Python, Algorithms** - 🔗 </>

- ◦ Implemented a tool converting C++ functions to LaTeX pseudo-code, reducing time spent on professional writing by over 90%.
- ◦ Won the IIT Indore Hackathon.

Dynamic Instructions Calculation **- Python, Flask, Bash** - 🔗 </>

- ◦ Devised a **GDB**-based bash and python **script** to get instruction analysis for a given program with 100% precision.
- ◦ Implemented **REST API** service with the **script** on the **server** to find instructions analysis of the first 10000 instructions.

Sampark **- NodeJS, ReactJS** - 🔗 </>

- ◦ Developed a peer-to-peer 2-way video chat using **WebRTC** & **SocketIO** web application for real-time video communication.
- ◦ Selected as top 23 in NIT-Rourkela Hackathon among 5000+ participants.

3.3 What is your experience with Python and Javascript?

I have experience in Python as well as JavaScript programming. I got introduced to Python and JavaScript in my first year and have been using them extensively in my projects for the last three years. I have made the projects as shown above on the above technology extensively. All the projects are deployed and some of them are used in production. Also I have had volunteer work experience which are as follows

Computer Programming Tutor - RGIPT

- Taught programming in **C** and **Python** to over 57 junior students, guided in solving **100+** problems of basic Algorithms.

Technical Head - ACM - SC

- Automated bulk emailing using python scripts and designed the official website, leading to **99%+** time savings.
- Lectured on **Open Source Development, Git,** and **GitHub** and tutored workshops for over 100 attendees.

4 CloudCV Contributions

4.1 Pull Requests

- Fix #3884: Improve Validate Challenge Config to handle corner cases #3890

  Impact: This script handles every corner case which could happen while creating a challenge.

  (+97 -58 Lines)
- Enhancement: Add a Script to create challenges locally #71

  Impact: The script would help in testing, validating, and creating challenges locally which would help contributors to facilitate better testing and evaluation.

  (+298 Lines)
- Fix #342: Fix error on get all challenges #344

  Impact: This fixed the major issues in eval ai cli where we were not able to get challenges.

  (+3 -3 Lines)

4.2 Issues

- Documentation: Host challenge: Fix duplication 3913

5 Other open-source contributions

- Kernel Adiutor: Fix Temperature Calculation bug. #532
- Kernel Adiutor: ThunderCharge: Fix Charge current not changing bug. #518

AR_Beast Kernel **- Linux Kernel, Android, App, C** - Open Source 🔗 </>

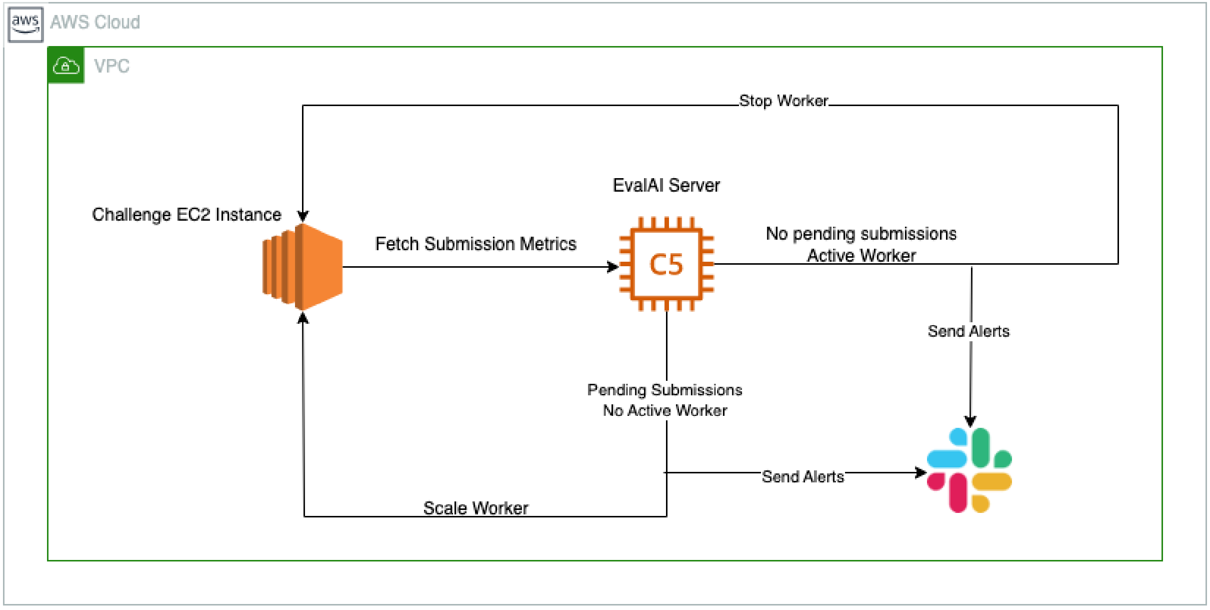
- Built Custom Linux Kernel for 4 Android Devices as an open source project, which helped **9000+** users ↗ to get an optimized device experience.
- Coded device-specific Thermal control driver 📄 & Current Control driver 📄 in C, which reduced heating, increased performance by **10**% and improved charging time by **30** min.

- Overhauled Android App </> to easily configure over 50 features offered by the kernel, reaching over **30**,**000** downloads ↗.

6 Project Details

6.1 Project Description

EvalAI is a growing platform that requires performance optimization. Currently, the system relies on admins for challenge hosting on code upload-based challenges, where the EvalAI team sets up the infrastructure. This dependence on admins is not ideal, and there is a need to automate large worker deployments in AWS to make challenge management more efficient. Further development of a comprehensive and efficient test suite will make the system more robust and prevent code-breaking changes.



6.2 Deliverables:

1. **Infrastructure Optimization:** The first objective is to automate large worker deployments on EvalAI using AWS EC2 instances or spot instances. This will reduce the dependency of challenge hosts on EvalAI admins and make challenge management more efficient. The infrastructure optimization will include the following tasks:

   (a) Implement auto-scaling for the challenges being hosted on independent EC2 instances.

   (b) Add a feature to allow hosts to use custom docker images based on EvalAI worker images for submission evaluation.

   (c) Add features to create a forum for the challenge during challenge creation.

   (d) Add a feature in GitHub-based challenge to manage multiple challenge configs over the years for the same challenge.

2. **Building a Test Suite:** The second objective is to make EvalAI more reliable and error-free by incorporating tests for different frontend and backend components. This will prevent code-breaking changes to the codebase and enhance the overall quality of the platform. The test suite development will include the following tasks:

   (a) Add tests for GitHub-based challenge creation on EvalAI.

(b) Add tests for code upload evaluation workers, including unit tests for individual components and integration tests for the worker.

(c) Add tests for code-upload challenge evaluation and static code-upload challenge evaluation pipelines.

(d) Add unit tests of Kubernetes components using mock.

(e) Add tests for frontend components:

   i. Challenge Page
   ii. Make submission page
   iii. My submissions and All submissions page
   iv. Settings tab
   v. Dashboard with tabs for all challenges hosted challenges and participated challenges.

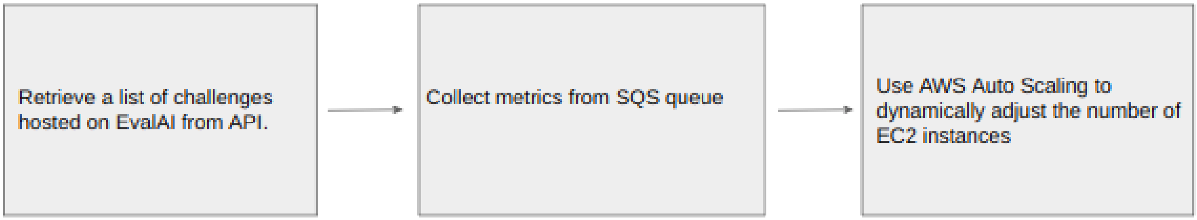
(f) Integrate tests

7 Project Implementation

7.1 Infrastructure Optimization

7.1.1 Implement auto-scaling for the challenges being hosted on independent EC2 instances

To enhance the scalability of EvalAI and accommodate the changing workload of hosting challenges, we implement auto-scaling for the independent EC2 instances hosting these challenges. This will involve designing a scalable architecture that can handle failures and ensure high availability while supporting auto-scaling for both EC2 and spot instances. To achieve this goal, we will proceed as follows:


Retrieve a list of challenges hosted on EvalAI from API.
Collect metrics from SQS queue
Use AWS Auto Scaling to dynamically adjust the number of EC2 instances

1. Obtain a list of challenges hosted on EvalAI by utilizing the API endpoint `api/challenges/challenge/all/all/all`. This will provide us with a list of challenges that need to be monitored for load conditions.

2. Gather metrics from the Simple Queue Service (SQS) queue for each challenge and monitor the queue length to determine the number of messages in each SQS queue. This will enable us to assess the load conditions for each challenge.

3. Scale up or down the number of EC2 instances hosting each challenge based on the load conditions inferred from the queue length. This will be accomplished through the use of AWS Auto Scaling, which will allow us to scale the number of instances dynamically as needed. We can also implement alerts to notify us when workers are starting or ending to facilitate effective management of the system. With this approach, we will be able to effectively monitor the load conditions for each challenge and scale the number of instances as needed, ensuring optimal performance and availability of the system.

7.1.2 Add the feature to allow hosts to use custom docker images based on EvalAI worker images for submission evaluation

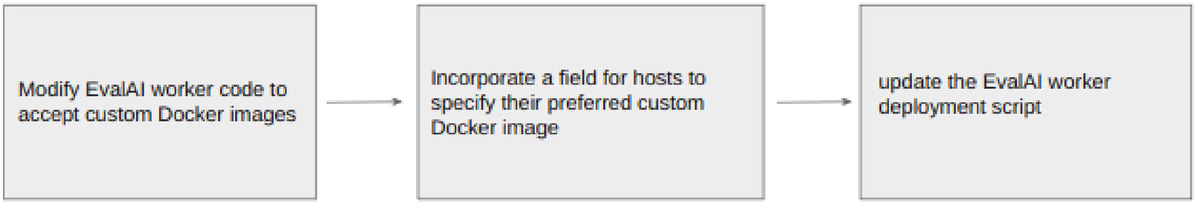
To enhance the functionality of EvalAI and provide greater flexibility to hosts, we add a new feature allowing hosts to utilize custom Docker images based on EvalAI worker images for submission evaluation.

To achieve this objective, we will implement the following steps:

1. Modify the EvalAI worker code to accept custom Docker images. This will involve updating the EvalAI API to permit hosts to specify custom Docker images for submission evaluation. To accomplish this, we will create a new API endpoint that accepts the Docker image parameter.

2. Modify the EvalAI challenge creation process to incorporate a field for hosts to specify the custom Docker image they want to use for submission evaluation. This field will be included in the GitHub create challenge configuration.

3. Update the EvalAI worker deployment script to pull the custom Docker image specified by the host and use it for evaluation purposes.


Modify EvalAI worker code to accept custom Docker images
Incorporate a field for hosts to specify their preferred custom Docker image
update the EvalAI worker deployment script

7.1.3 Add features to create a forum for the challenge during challenge creation

To implement the feature of creating a forum on the challenge page to send messages directly to the hosts during challenge creation, we would follow the below steps:

1. During the challenge creation, a new boolean variable ”create_forum” would be added along with the email. This variable will indicate whether to create a new section on the challenge page for sending messages to the hosts or not.

2. Once the challenge is created, a new section on the challenge page for the forum would be created using the appropriate design on the front end in Angular. This section would have the details for contacting the hosts, such as the email address of the hosts, subject, and message.

3. The form for sending the email to the hosts would use EvalAI's email address. To achieve this, the following steps would be taken:

   (a) A new email template for sending messages from EvalAI's email address would be created. This template would include the necessary details, such as the sender's name and email, the challenge name, and the message content.

   (b) A third-party email service like SendGrid would be used to send the email. The SendGrid API would be used to send the email by providing the email content and recipient details.

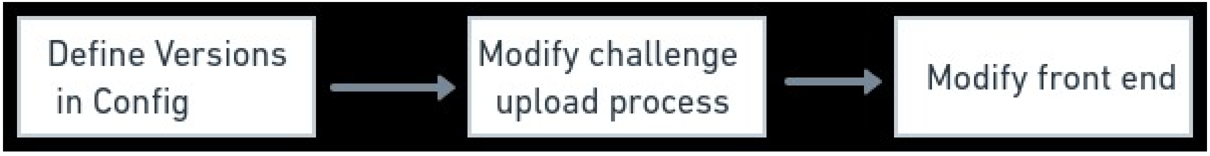
   (c) Appropriate email headers and content would be added to ensure the email is delivered successfully. This could include setting the ”From” and ”Reply-To” fields, adding relevant subject lines, and including appropriate email content.

4. Finally, the pre-defined email template would be used to generate the email content dynamically before sending the email to the host. This would ensure that the email includes all the necessary details and follows a consistent format.

By following these steps, a section on the challenge page to send messages directly to the hosts would be implemented, using a pre-defined email template. This would help users to communicate with the hosts and get their queries resolved promptly.

7.1.4 Add a Feature in GitHub-based Challenge to Manage Multiple Challenge Configs

The current GitHub-based challenge configuration allows us to modify and update the configs by pushing to the repository. To enhance this feature, we can add the ability to manage multiple challenge configurations for the same challenge over the years. To implement this feature, we follow the following steps:



1. Define a new variable: Define a new variable named ”versions” in the GitHub repository’s configuration file that keeps track of the different versions of the challenge configuration over time. This variable can be an array with each element representing a version of the configuration.

2. Update the challenge upload process: Modify the challenge upload process to include a version number and update it when uploading a new challenge configuration. This can be done by adding a field to the upload form where the user can specify the version number.

3. Update the front end: Modify the front end to display the most recent version of the challenge configuration. This can be achieved by retrieving the latest version number from the ”versions” array in the repository’s configuration file and displaying it on the front end.

7.2 Building a Test Suite

The second major goal of the project is to enhance the reliability and error-free operation of EvalAI. To achieve this, it is crucial to incorporate tests for various frontend and backend components. By adding robust tests, we can prevent code-breaking changes to the codebase. To accomplish this goal, we can follow these steps:

1. Identify the different API endpoints, the pipeline of prediction upload evaluation workers, and code upload evaluation workers that require unit testing.

2. Write test cases for each endpoint and worker using the pytest testing framework.

   Pytest is a popular Python testing framework that allows developers to write clear and concise test cases.

3. A directory called ’tests’ is present in the root directory of the project. Inside the ’tests’ directory, create a file called ’test_¡component¿.py’ for each component that requires testing. For example, ’test_api.py’ for API endpoint testing, ’test_workers.py’ for worker testing, and so on.

4. Write test cases for each component using the pytest framework, ensuring that you cover both success and failure scenarios. For example, in GitHub-based challenge creation, we would handle cases of failing scenarios like some major required field is missing. Similarly, we would add each component.

For each test case, include descriptive names, comments, and assertions to ensure that the test cases are clear and understandable.

5. For integration testing, we need to create a separate test environment and write integration tests to ensure that all components interact correctly with each other.
    (a) Create a new environment for testing using tools like Docker or virtual environments.
    (b) In the new environment, install the dependencies required for testing.
    (c) Write integration tests that simulate the interaction between different components of the application.
    (d) Use the Pytest framework to run integration tests alongside unit tests.

By following this approach, we can add tests for the following components:

7.2.1 Add tests for GitHub-based challenge creation on EvalAI.

7.2.2 Add tests for code upload evaluation workers, including unit tests for individual components and integration tests for the worker.

7.2.3 Add tests for code-upload challenge evaluation and static code upload challenge evaluation pipelines.

7.2.4 Add unit tests for Kubernetes components using a mock environment

When adding unit tests for Kubernetes components, it is important to simulate a Kubernetes cluster in a mock environment to avoid making any actual changes to the cluster. This can be done using the Kubernetes testing framework, which provides a way to create a mock environment that simulates the Kubernetes API server, etc database, and other Kubernetes components.

To add unit tests for Kubernetes components in EvalAI, we can follow these steps:

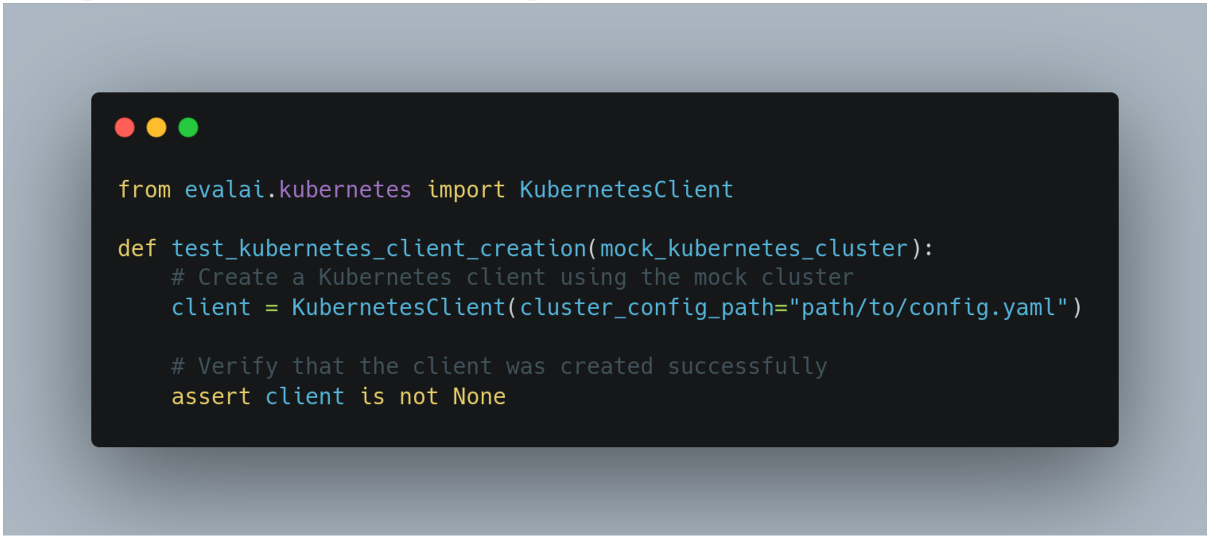
1. Install the Kubernetes testing framework: The Kubernetes testing framework is a Python package that can be installed using pip. We can install it in the same virtual environment as the rest of the EvalAI project.

2. Create a mock Kubernetes cluster: We can use the Kubernetes testing framework to create a mock Kubernetes cluster in a test function. This mock cluster will simulate the Kubernetes API server, etc database, and other Kubernetes components, allowing us to test our Kubernetes components in a controlled environment.

3. Write test cases using the Pytest framework: We can use the Pytest framework to write test cases for each Kubernetes component in EvalAI. These test cases should cover both success and failure scenarios, testing various edge cases and error conditions.

4. Test Kubernetes components: We can run the test cases using the Pytest framework to ensure that they pass and do not cause any errors. The test cases should verify that the Kubernetes components are working correctly and that they can handle various types of input and output.

Example test case for Kubernetes component:



```
from evalai.kubernetes import KubernetesClient

def test_kubernetes_client_creation(mock_kubernetes_cluster):
    # Create a Kubernetes client using the mock cluster
    client = KubernetesClient(cluster_config_path="path/to/config.yaml")

    # Verify that the client was created successfully
    assert client is not None
```

In this example, we are testing the creation of a Kubernetes client using a mock cluster. The mock_kubernetes_cluster fixture is created using the Kubernetes testing framework, and it provides a mock cluster for testing. We create a Kubernetes client using the KubernetesClient class from the evalai.kubernetes module and verify that it is not None. This test case verifies that the Kubernetes client can be created successfully in a mock environment.

7.2.5 Add tests for frontend components

To achieve testing here we would use Karma with Jasmine, which is a testing framework that runs in the background and allows you to write automated tests for your application in a headless browser. Karma is designed specifically for testing Angular applications and can be integrated with other tools for end-to-end testing. Jasmine is the testing framework that is typically used with Karma, and it provides a clean syntax for writing tests that are easy to read and understand.

In the case of EvalAI, since the front end is written in Angular, it would be more appropriate to use Karma with Jasmine for unit testing and integration testing of the front-end components. Protractor could also be used for end-to-end testing of the entire application, but Karma with Jasmine is a more appropriate choice for testing individual components of the Angular front.

To add tests for frontend components in EvalAI using Angular, we can follow these steps:

1. Identify the different front-end components of the application, including buttons, forms, and pages.

2. Using the testing framework for Angular, Karma and Jasmine, to write automated tests that simulate user interaction with the frontend components. Karma is a test runner for JavaScript-based applications, and Jasmine is a testing framework for JavaScript-based applications. We can install these tools using the following commands: We can then write test cases and save them in a separate tests/ directory within the Angular project.

3. For integration testing, we need to create a separate test environment and write integration tests to ensure that all components interact correctly with

each other. We can use tools like Protractor for end-to-end testing of Angular applications. Protractor is an end-to-end testing framework for Angular and AngularJS applications. We can install Protractor using the following command: We can then write end-to-end test cases in TypeScript and save them in a separate tests/ directory within the Angular project.

4. Run the tests to ensure that they pass and do not cause any errors. We can run the tests using the following command: This will run all the test cases in the tests/ directory. We can also specify a specific test file or test case to run using the command: For testing frontend components that interact with the backend, we can use tools like ngMock to mock the API calls and test the frontend components in isolation.

By following this approach, we can add tests for various frontend components in EvalAI, including:

- Challenge Page
- Make Submission Page
- My Submissions and All Submissions Page
- Settings tab
- Dashboard with tabs for all challenges, hosted challenges and participated in challenges.

7.2.6 Integrate tests

We would add all the testing to the pipeline tool Travis CI which is used on EvalAI to automate running the tests.

1. Ensure that the pipeline is set up to run both unit tests and integration tests.
2. Configure the pipeline to report test results and notify developers of any test failures or errors.

Overall, adding unit tests for Kubernetes components using a mock environment can help ensure that the Kubernetes components in EvalAI are working correctly and can handle various types of input and output. By covering both success and failure scenarios, we can ensure that the Kubernetes components are reliable and can handle any unexpected errors that may arise.

8 Project Schedule

8.1 GSoC period

Period	**Duration**	**Goals**
Community Bonding	May 4 - May 28	- Explore EvalAI's codebase to understand its different components and Discuss project requirements with mentors to better understand their expectations. - Learn relevant skills, which I would need to for setting up testing frameworks and DevOps
Week 1	May 29 - Jun 4	- Set up auto-scaling for challenges hosted on independent EC2 instances
Week 2	Jun 5 - Jun 11	- Add feature to allow hosts to use custom docker images based on EvalAI worker images for submission evaluation and buffer time for any left work
Week 3	Jun 12 - Jun 18	- Buffer Period - Start working on forum feature for challenge creation
Week 4	Jun 19 - Jun 25	- Start working on multiple challenge configs feature for GitHub-based challenge
Week 5	Jun 26 - Jul 2	- Buffer Time for anything left - Start building a test suite for GitHub-based challenge creation on EvalAI.
Week 6	Jul 3 - Jul 9	- Continue Building test suite for GitHub based challenge creation - Add tests for code-upload challenge evaluation and static code-upload challenge evaluation pipelines.
Midterm Evaluation	Jul 10 - Jul 14	- Submit midterm evaluations
Week 7	Jul 15 - Jul 21	- Add tests for code upload evaluation workers, including unit tests for individual components and integration tests for the worker.
Week 8	Jul 22 - Jul 28	- Buffer Time for anything left - Add unit tests of Kubernetes components using mock.
Week 9	Jul 29 - Aug 4	- Add tests for frontend components: all page
Week 10	Aug 5 - Aug 11	- continue to Add tests for frontend and buffer for anything left.
Week 11	Aug 12 - Aug 18	- Finalise any pending work and Ensure the project is well-documented and tested.
Final Evaluation	Aug 21 - Aug 28	- Submit final work product and mentor evaluation

8.2 Pre GSoC period

To continue contributing to EvalAI, I plan to focus on fixing issues, adding new features, and improving the codebase. I will begin by ensuring that any outstanding pull requests I have submitted are fully addressed and merged. If any further action is required, I will communicate with my mentors. Furthermore, I plan to contribute to the project by taking on new challenges and exploring areas of the codebase that I am less familiar with. This will allow me to expand my knowledge and skills, as well as provide valuable contributions to the project. I would take the opportunity to work on new issues and take on more responsibilities within the project.

8.3 Post GSOC period

Contributing to open source is not just something I am doing for GSOC. It has been a valuable experience for me to solidify my skills and help others. I am passionate about expanding my knowledge and sharing it with others. Even after the GSOC period ends, I will ensure that the projects I have implemented have proper documentation for knowledge transfer. I plan to continue contributing to this project and making the platform better.

9 Time

I am fully dedicated to devoting my complete focus and effort to the project from start to finish. During my summer break, from May 5th to August 31st, which is the complete period for GSoC I have no prior engagements or internships, which allows me to commit more than 40 hours per week to the project. I am enthusiastic about investing my time and energy into the project to ensure its successful completion. My usual work hours are from 11 AM to 2 AM (Indian Standard Time), but I am flexible and can adjust my schedule to accommodate the project's needs. Furthermore, I am open to spending additional time learning new skills and tackling any obstacles that may arise during the project. I have already allocated ample time to develop a comprehensive plan, including buffer days, and to acquire any necessary skills to ensure that I remain on track and deliver high-quality work. However, I am willing to devote extra hours to ensure the project's timely completion and exceed expectations.

References

[1] Cloud-CV GSoC Ideas. GitHub. Retrieved April 4, 2023, from https://github.com/Cloud-CV/GSoC-Ideas/issues/60.

[2] Cloud-CV GSoC 2023 Proposal Template. GitHub. Retrieved April 4, 2023, from https://github.com/Cloud-CV/GSoC-Ideas/wiki/GSOC-2023-Proposal-Template.

[3] EvalAI Documentation. Retrieved April 4, 2023, from https://evalai.readthedocs.io/.