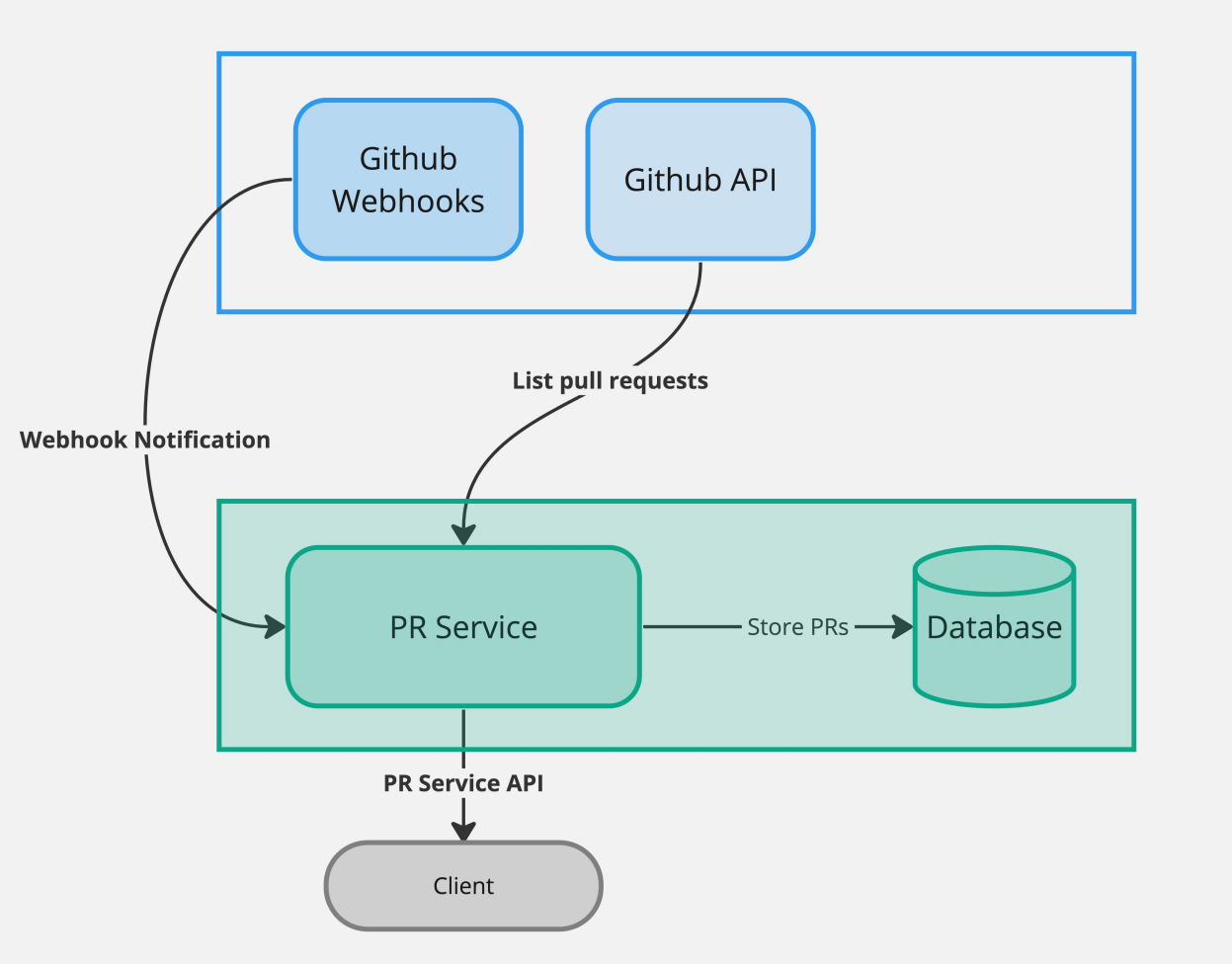
AvantStay Bookings & Distributions Challenge

Github Challenge



The application is designed to interact with **pull request events** through webhooks, ensuring idempotent processing to handle duplicate events without

inconsistencies. It also includes a **scheduled job** that fetches pull request data twice daily.

Additionally, the application features an API to **deliver metrics related to the pull requests**, focusing on scalability, security, and efficiency in data handling and retrieval.

The **service API** must provide the following metrics.

Project Metrics

1. **Total Contributors**: The total number of unique contributors who have made at least one commit to the project. This reflects the diversity and size of the project's developer community.
2. **Total Commits**: The cumulative number of commits made to the project. This indicates the project's level of activity and development progress.
3. **Total Closed Pull Requests**: The total number of pull requests that have been closed, including both merged and unmerged requests. This metric helps understand how many proposed changes or contributions have concluded.
4. **Total Open Pull Requests**: The number of pull requests currently open. This shows the amount of ongoing work or pending contributions to the project.

Contributor Metrics

1. **Total Projects**: The number of different projects a contributor has made commits to. This reflects the breadth of a contributor's involvement across various projects.
2. **Total Commits**: The total number of commits a contributor has made across all projects they've contributed to. This indicates the contributor's level of activity and commitment.
3. **Total Closed PRs**: The number of pull requests the contributor has closed (either by having their contributions merged or otherwise). This metric shows how many of the contributor's proposed changes have been finalized.
4. **Total Open PRs**: The number of pull requests initiated by the contributor that are still open. This indicates the contributor's ongoing contributions and pending work.

These metrics collectively provide insights into the *activity*, *participation*, and *progress* within a project or a contributor's engagement across projects.

Requirements

1. **Receive and process webhook notifications** for pull request events
    - It must be idempotent, the system needs to be able to process duplicated events
2. **Create a job** to periodically get pull requests twice a day
3. **Create API** to return the metrics

> 💡 You are free to adapt the requirements as necessary to reach the goal

Service API

1. *Get project metrics*

```
GET /projects/:projectId/metrics

// Response example
{
  "totalContributors": 0,
  "totalCommits": 0,
  "totalClosedPRs": 0,
  "totalOpenPRs": 0
}
```

2. *Get contributor metrics*

```
GET /contributors/:contributorId/metrics
// Response example
{
  "totalProjects": 0,
  "totalCommits": 0,
  "totalClosedPRs": 0,
  "totalOpenPRs": 0
}
```

👌 We can create an **Restful API** or **GraphQL API** to provide the above queries

Tech Stack

You can choose any Scala library for the project and any database.

Desirable is to have the project covered by **Typelevel environment** and **Cats Effect 3**.

Tips

1. Use Ngrok to configure Github webhook to receive the notification on your localhost
2. You can test the project in a sample Github repository to trigger the events
3. You can configure database in Docker to execute the project
4. You can use Kafka as the event bus to receive and then process the events. Feel free to pick other solution or tool, such as Redis or an in-memory queue
5. We appreciate well written code
6. Add necessary test you consider useful - use any Scala test tool
7. Runnable project is very appreciated

8. Create a project in Gitlab or Github to share it with us

References

- RestAPI Pull Requests
- Creating webhooks
- Pull Request events
- NGROK
- Typelevel Platforms