THE APACHE® SOFTWARE FOUNDATION

Google
Summer of Code

Divyansh Khatri

Major: Information Technology

University:Bhagwan Parshuram Institute of Technology,Delhi

Github: @Divyansh200102

Email: divyanshkhatri200102@gmail.com

LinkedIn: @divyansh-khatri-4402bb20

Phone: (+91) 7303660054

Timezone: Indian Standard Time (UTC +5:30)

Apache ShenYu KitexPlugin

About me

I am currently a second-year undergraduate majoring in Information Technology at Bhagwan Parshuram Institute of Technology,Delhi. I completed my high school education at St. Gregorios School,Delhi .I demonstrate a strong work ethic in my academic pursuits and maintain a fervent eagerness for learning. Within my college, I actively participate in technical as well as non-technical activities. Additionally I am a part of GDSC through which I have had the privilege of expanding my understanding of Google's developer technologies and collaborating with peers on innovative projects.

Background

Existing System and Identified Need

Existing:

- **Apache ShenYu:** A Java-based API Gateway offering service proxy, protocol conversion, and API governance functionalities. While strong in Java, its support for multiple languages is limited.
- **WebAssembly (Wasm):** An efficient binary format for code designed for execution across various platforms at near-native speeds.

- **Kitex:** A popular Go RPC framework used for building microservices.

Identified Need:

ShenYu currently lacks robust support for developing custom plugins in languages beyond Java. This restricts developers who might prefer languages like Go or Python for plugin creation.

- Extend ShenYu's plugin functionality beyond Java to support Kitex microservices.
- Leverage the efficiency and portability benefits of Wasm for custom ShenYu plugins.

Reuse and Replacement Strategy

Reuse:

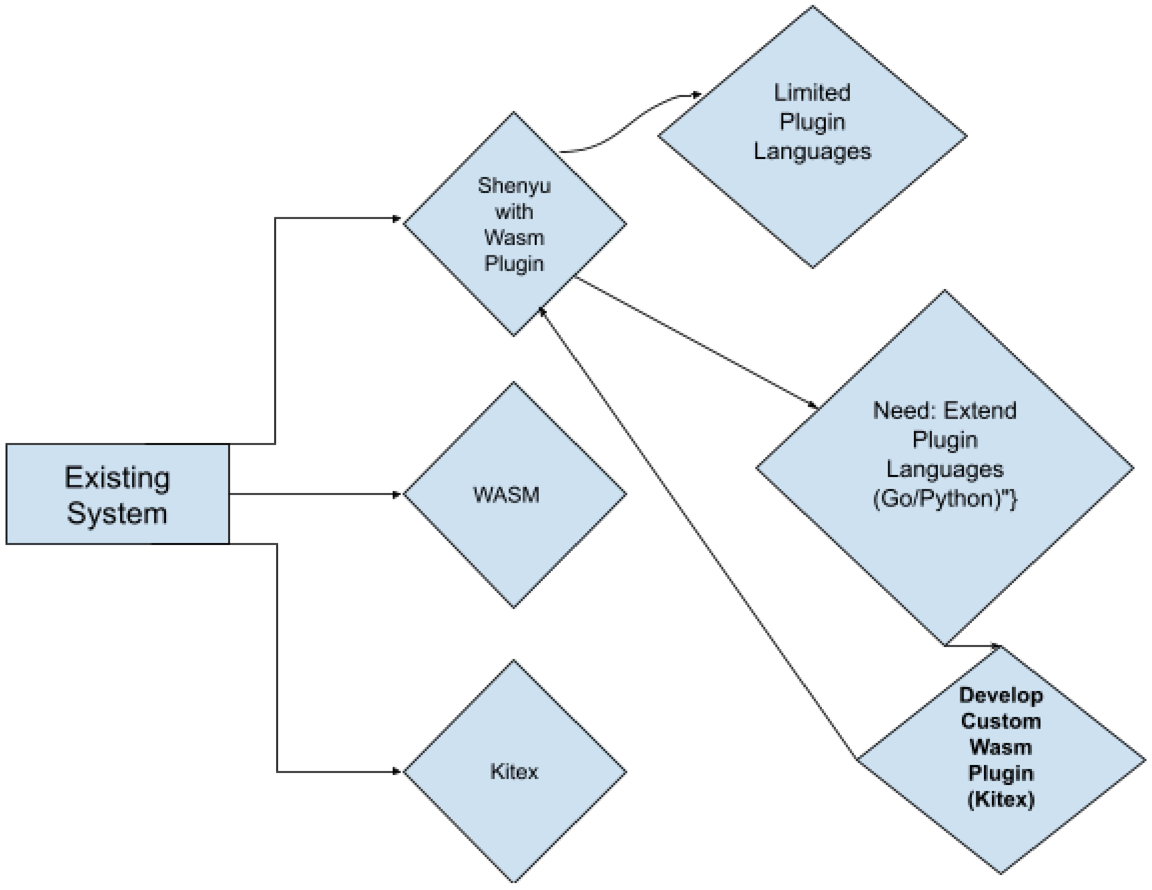
- **ShenYu Core Functionalities**: The core functionalities of ShenYu, including service proxy, protocol conversion, and API governance, can be entirely reused in this project.
- **ShenYu Wasm Plugin Feature:** This existing functionality within ShenYu enables loading and execution of Wasm plugins, eliminating the need to rewrite core gateway logic.

Replacement:

- **ShenYu Kitex Plugin Implementation:** The current, potentially limited, implementation within ShenYu will be replaced with a custom Wasm plugin. This new plugin will be developed using WebAssembly and leverage Kitex functionalities.
- The backend code responsible for loading and executing plugins will need to be replaced or extended to accommodate the execution of WebAssembly bytecode. Additionally, the project may involve creating new documentation and testing frameworks tailored to the integration of Kitex with Apache ShenYu.

Justification:

This initiative proposes the integration of the Wasm Plugin feature into Apache ShenYu, enabling developers to leverage WebAssembly for crafting custom plugins. By doing so, it circumvents the language constraints previously imposed on ShenYu plugin development. This integration capitalizes on the performance and cross-platform portability offered by WebAssembly, while retaining the core capabilities and plugin framework of ShenYu.


Existing System
Shenyu with Wasm Plugin
WASM
Kitex
Limited Plugin Languages
Need: Extend Plugin Languages (Go/Python)"}
Develop Custom Wasm Plugin (Kitex)

Community engagement

In my past contributions, I have demonstrated proficiency in working with WebAssembly (Wasm) technology by adding test cases for Wasm files, showcasing my experience and understanding in this domain. Specifically, I have contributed to Apache ShenYu by adding test cases for critical components such as:

Add test cases for the **Shenyu-Plugin-Wasm-Base module.**

- **Add test case for AbstractWasmPluginDataHandler(#5451)**
- **Add test case for AbstractWasmPluginDiscoveryHandler(#5453)**

Moreover, within the Apache ShenYu project, I have extensively contributed to enhancing test coverage across various modules. Notable contributions include:

- **Add test case for ShenyuClientRegisterRepositoryFactory (#5443)**
- **Add test case for ShenyuClientRegisterEventPublisher (#5417)**
- **Add test case forShenyuClientIllegalArgumentException (#5408)**
- **Add test case for ShenyuClientMetaDataExecutorSubscriber (#5404)**
- **Shenyu-Client-Motan test case (#5402)**

Furthermore, beyond ShenYu, I have actively engaged with the Apache Software Foundation community, through my contributions

Apache Arrow

MERGED PR'S

- **GH-39291: [Docs] Remove the "Show source" links from doc pages (#40167)**
- **GH-40145: [C++][Docs] Correct the console emitter link (#40146)**
- **GH-39416: [GLib][Docs] Fixed Broken Link in README Content (#39896)**
- **GH-36044: [Python][Docs] Added ParquetFileFragment to the API reference docs (#38277)**
- **GH-35369: [Docs] Fix reST markup around ref:IPC format <format-ipc> (#38276)**
- **GH-38216: [R] open_dataset(format = "json") not documented (#38258)**

OPEN PR'S

- **GH-31315: [C++][Docs] Document that the strptime kernel ignores %Z (#40186)**
- **GH-39759: [Docs] Update pydata-sphinx-theme to 0.15.x (#39879)**
- **GH-33745: [Docs] Updated "struct_field" kernel documentation (#39851)**
- **GH-31538: [Python][Docs] Documents ParquetWriteOptions class (#38279)**

Apache BRPC

MERGED PR'S

- **Update cases.md (#2540)**
- 更新**brpc**在各个公司的落地场景 **(#2464)**
- **remove the docs and comments related to HealthCheckThread (#2463)**
- **Remove implementation specific function name of EventDispatcher (#2462)**
- **changed long to size_t in AnnotateBenignRaceSized function (#2403)**

Regarding engagement with the community, I have actively participated in project mailing lists, wikis, and issue trackers. I have initiated discussions, sought feedback, and collaborated with other community members to refine proposals, resolve issues, and contribute to the collective knowledge base.

As part of my project plan, I intend to incorporate user testing, prototypes, and thorough code reviews into the development process. Prototypes will be utilized to iteratively design and refine solutions before full-scale implementation. Code reviews will ensure the quality, maintainability, and adherence to project standards of all contributions.

Design / description of work

This section outlines the design and development efforts to integrate Apache ShenYu with the Kitex RPC framework.

1. Enhance ShenYu Client with Kitex Support:

- **Task:** Integrate the Kitex RPC framework into the existing `shenyu-client-golang` library.
- **Description:** This task involves extending the `shenyu-client-golang` library to seamlessly interact with services built using the Kitex framework. This will enable developers to leverage ShenYu's gateway and rule engine functionalities for their Kitex-based microservices.

2. Develop ShenYu Plugin for Kitex:

- **Task:** Create a new ShenYu plugin specifically designed for handling Kitex service invocations.
- **Description:** This task requires building a custom plugin within the ShenYu ecosystem to intercept and manage requests targeting Kitex services. The plugin will enable Apache ShenYu to route, proxy, and manage requests to microservices built with Kitex, leveraging the high-performance and extensible features of the Kitex framework.

3. Simplify Integration with Spring Boot:

- **Task:** Introduce a `shenyu-spring-boot-starter-plugin-kitex` module to streamline integration with Spring Boot applications utilizing Kitex services.
- **Description:** This task focuses on creating a pre-configured Spring Boot starter module that simplifies the process of incorporating ShenYu's Kitex plugin into Spring Boot projects. This module will automatically configure necessary dependencies and enable seamless integration of Kitex-powered microservices with Apache ShenYu's plugin system, enabling developers to easily configure and manage routing and proxying of requests within their Spring Boot applications.

4. Establish Comprehensive Test Suite:

- **Task:** Develop a dedicated `shenyu-integrated-test-kitex` module for comprehensive testing of the Kitex integration functionalities.
- **Description:** This task involves building a comprehensive test suite specifically designed to validate the functionality and behavior of ShenYu when interacting with Kitex services. This will ensure the reliability and robustness of the integration and catch potential issues before deployment.

5. Document and Publish Knowledge:

- **Task:** Contribute to the ShenYu website by creating detailed documentation for the newly developed Kitex plugin.
- **Description:** This task requires creating clear and informative documentation on the ShenYu website to guide developers on utilizing the Kitex plugin effectively. This documentation will cover installation, configuration, usage examples, and best practices for integrating ShenYu with Kitex-based services.It will serve as a comprehensive resource for developers seeking to leverage the capabilities of Kitex within the Apache ShenYu ecosystem, offering clear guidelines and examples for implementation and deployment.

By completing these tasks,Apache ShenY will be seamlessly integrated with the popular Kitex RPC framework, enhancing its flexibility and expanding its reach within the microservices ecosystem.

Deliverables

This proposal outlines the deliverables for integrating Apache ShenYu with the Kitex RPC framework, aiming to enhance ShenYu's capabilities and simplify development processes.

1. Enhanced ShenYu Client with Kitex Support:

- **Deliverable:** The `shenyu-client-golang` library will be extended to seamlessly interact with Kitex services. This will enable developers to leverage ShenYu's gateway and rule engine functionalities for their Kitex-based microservices.

2. ShenYu Plugin for Kitex:

- **Deliverable:** A custom ShenYu plugin specifically designed for handling Kitex service invocations will be developed. This plugin will enable Apache ShenYu to route, proxy, and manage requests to microservices built with Kitex.

3. Simplified Integration with Spring Boot:

- **Deliverable:** A pre-configured Spring Boot starter module named `shenyu-spring-boot-starter-plugin-kitex` will be created. This module will simplify the process of incorporating ShenYu's Kitex plugin into Spring Boot projects by automatically configuring necessary dependencies and enabling seamless integration with Kitex-powered microservices.

4. Comprehensive Test Suite:

- **Deliverable:** A dedicated `shenyu-integrated-test-kitex` module will be developed for comprehensive testing of the Kitex integration functionalities. This module will ensure the reliability and robustness of the integration.

5. Documentation and Knowledge Sharing:

- **Deliverable:** Detailed documentation for the newly developed Kitex plugin will be created and published on the ShenYu website. This documentation will cover installation, configuration, usage examples, and best practices for integrating ShenYu with Kitex-based services.

Overall Impact:

By successfully completing these deliverables, ShenYu will be seamlessly integrated with the popular Kitex RPC framework, extending its flexibility and reach within the microservices ecosystem. This integration will empower developers to leverage the combined capabilities of both frameworks, fostering a more efficient and robust development experience.

Weekly Scheduling (12 Weeks)

This schedule outlines the development plan for the Shenyu Kitex plugin within a 3-month GSoC timeline, considering key milestones and incorporating technical considerations.

Community Bonding Period (May 1 - 26)

- **Week 1-2:**
    - Familiarize myself with the Shenyu project by reviewing documentation (https://shenyu.apache.org/) and exploring the codebase, particularly the existing client and plugin modules (e.g., https://github.com/goauthentik/client-go).
    - Establish communication with my mentors and actively participate in community discussions to gain insights and ask questions.
    - **Deliverable:** Gain a solid understanding of the Shenyu architecture and Kitex integration requirements.

- **Week 3-4:**
    - Deep dive into the Kitex framework and its interaction with gRPC. Understand how Kitex defines services, requests, and responses.

    - Begin planning the Shenyu Kitex client implementation (`shenyu-client-kitex`) by outlining the integration approach with the existing `shenyu-client-golang` module.
    - **Deliverable:** Develop a solid understanding of Shenyu Kitex client architecture and integration strategy.

Coding Period (May 27 - August 26)

(**Note**:*The code snippets presented in this section represent preliminary implementations of the structure and functionality. These implementations are subject to further development and refinement as the project progresses.*)

Week 1-2 (May 27 - June 9):

- **Focus:** Shenyu Kitex Client Development (`shenyu-client-kitex`)
    - Implement core functionalities of the `shenyu-client-kitex` module, enabling interaction with Kitex services through Shenyu.
    - Ensure proper serialization and deserialization of data between Shenyu and Kitex.
    - Conduct unit testing to validate the client's functionality.
    - **Deliverable:** Functional `shenyu-client-kitex` module with unit tests.

```go
// (Conceptual snippet - Shenyu Kitex Client)

type ShenyuKitexClient struct {
    client   shenyu.Client
    codec    codec.Codec
    reqCodec codec.Codec
    resCodec codec.Codec
}

func NewShenyuKitexClient(client shenyu.Client, codec codec.Codec, reqCodec,
resCodec ...codec.Codec) (*ShenyuKitexClient, error) {
    if client == nil {
        return nil, errors.New("shenyu client cannot be nil")
    }
    if codec == nil {
        return nil, errors.New("kitex codec cannot be nil")
    }
```

```
  c := &ShenyuKitexClient{
        client: client,
        codec:  codec,
    }

    if len(reqCodec) > 0 {
        c.reqCodec = reqCodec[0]
    }
    if len(resCodec) > 0 {
        c.resCodec = resCodec[0]
    }

    return c, nil
}

func (c *ShenyuKitexClient) Do(ctx context.Context, req *shenyu.Request)
(*shenyu.Response, error) {
    var kitexReq interface{}
    var err error

 if c.reqCodec != nil {
        kitexReq, err = c.reqCodec.Encode(req)
    } else {
        kitexReq, err = c.codec.Encode(req)
    }
    if err != nil {
        return nil, err
    }
    kitexResp, err := c.client.Do(ctx, kitexReq)
    if err != nil {
        return nil, err
    }

 var response *shenyu.Response
    if c.resCodec != nil {
        response, err = c.resCodec.Decode(kitexResp).(*shenyu.Response)
    } else {
        response, err = c.codec.Decode(kitexResp).(*shenyu.Response)
    }
    if err != nil {
        return nil, err
    }
    return response, nil
}
```

This code defines a client (`ShenyuKitexClient`) for interacting with **Kitex services** through **Apache ShenYu**.

Key components:

- **Shenyu Client:** Interacts with ShenYu gateway for service invocation.
- **Kitex Codec:** Serializes and deserializes data to/from Kitex format.
- **Optional Codecs (reqCodec, resCodec):** Handle custom serialization/deserialization for ShenYu requests and responses (if needed).

Functionalities:

- `NewShenyuKitexClient`: Creates a new `ShenyuKitexClient` instance.
- `Do`: Invokes a Kitex service via ShenYu with request/response handling:
  - Serializes request data using Kitex codec (or provided reqCodec).
  - Uses ShenYu client to send the request to the service.
  - Deserializes response data using Kitex codec (or provided resCodec).

Overall, this code acts as an adapter between ShenYu and Kitex, enabling clients to seamlessly call Kitex services through the ShenYu gateway.

Week 3-4 (June 10 - June 23):

- **Focus:** Shenyu Kitex Plugin Development (`shenyu-plugin-kitex`)
  - Design and develop the `shenyu-plugin-kitex` module, responsible for handling Kitex service requests within the Shenyu framework.
  - Integrate the `shenyu-client-kitex` module within the plugin for communication with Kitex services.
  - Implement routing logic within the plugin to map Shenyu requests to Kitex services.
  - Conduct unit testing to ensure the plugin's functionality.
  - **Deliverable:** Functional `shenyu-plugin-kitex` module with unit tests, integrated with `shenyu-client-kitex`.

```go
// (Conceptual snippet - Shenyu Kitex Plugin)

type ShenyuKitexPlugin struct {
    client   *ShenyuKitexClient
    matchers []shenyu.Matcher
}
```

```go
func NewShenyuKitexPlugin(client *ShenyuKitexClient, matchers ...shenyu.Matcher)
(*ShenyuKitexPlugin, error) {
    if client == nil {
        return nil, errors.New("shenyu client cannot be nil")
    }

    return &ShenyuKitexPlugin{
        client:   client,
        matchers: matchers,
    }, nil
}

func (p *ShenyuKitexPlugin) Match(ctx context.Context, req *shenyu.Request) bool {
    for _, matcher := range p.matchers {
        if matcher.Match(ctx, req) {
            return true
        }
    }
    return false
}

func (p *ShenyuKitexPlugin) Handle(ctx context.Context, req *shenyu.Request)
(*shenyu.Response, error) {
    if !p.Match(ctx, req) {
        return nil, errors.New("request does not match Kitex service")
    }

    // Delegate to ShenyuKitexClient for service invocation
    return p.client.Do(ctx, req)
}

func (p *ShenyuKitexPlugin) Route(ctx context.Context, req *shenyu.Request)
(*shenyu.Response, error) {
    // custom routing logic here (if needed)

    // If no custom routing is needed, delegate to Handle
    return p.Handle(ctx, req)
}
```

This code defines a plugin (`ShenyuKitexPlugin`) for Apache ShenYu that specifically handles requests targeting Kitex services.

Key components:

- `ShenyuKitexClient`: (Injected dependency) Client for interacting with Kitex services.
- `Matchers`: List of `shenyu.Matcher` objects used to identify Kitex requests based on defined criteria.

Functionalities:

- `NewShenyuKitexPlugin`: Creates a new `ShenyuKitexPlugin` instance.
- `Match`: Checks if an incoming ShenYu request matches a Kitex service using configured matchers.
- `Handle`: Invoked by ShenYu for matching requests. It delegates the actual service invocation to the `ShenyuKitexClient`.
- `Route` (Optional): Allows for implementing custom routing logic based on ShenYu rules and service discovery. If not implemented, it defaults to calling `Handle`.

In essence, this plugin acts as an intermediary within ShenYu. It identifies Kitex service requests and then leverages the `ShenyuKitexClient` to interact with the appropriate Kitex service.

Week 5-6 (June 24 - July 7):

- **Focus:** Shenyu Spring Boot Starter and Integration Tests (`shenyu-spring-boot-starter-plugin-kitex`, `shenyu-integrated-test-kitex`)
    - Develop the `shenyu-spring-boot-starter-plugin-kitex` module, simplifying integration of the `shenyu-plugin-kitex` with Spring Boot applications.
    - Create the `shenyu-integrated-test-kitex` module for integration and functional testing of the entire Shenyu Kitex plugin within a Spring Boot environment.
    - Conduct integration tests to verify the interaction between Shenyu, the Kitex plugin, and a simulated Kitex service.
    - **Deliverable:** Functional `shenyu-spring-boot-starter-plugin-kitex` and `shenyu-integrated-test-kitex` modules with integration tests.

```
@Configuration
public class ShenyuKitexAutoConfiguration {

    @Bean
    public ShenyuKitexPlugin shenyuKitexPlugin(ShenyuKitexClient client, List<ShenyuMatcher>
matchers) {
        return new ShenyuKitexPlugin(client, matchers);
    }

    @Bean
    public ShenyuKitexClient shenyuKitexClient(ShenyuClient shenyuClient, KitexCodec codec,
                                               ShenyuRequestCodec reqCodec,
ShenyuResponseCodec resCodec) throws Exception {
        if (shenyuClient == null) {
            throw new IllegalArgumentException("ShenyuClient cannot be null");
        }
        if (codec == null) {
            throw new IllegalArgumentException("KitexCodec cannot be null");
        }

        // Configure ShenyuKitexClient based on properties and dependencies
        return new ShenyuKitexClient(shenyuClient, codec, reqCodec, resCodec);
    }

    @Bean
    public ShenyuRequestCodec shenyuRequestCodec() {
        // Custom ShenyuRequestCodec logic implementation
        return new MyShenyuRequestCodec();
    }

    @Bean
    public ShenyuResponseCodec shenyuResponseCodec() {
        // Custom ShenyuResponseCodec logic implementation
        return new MyShenyuResponseCodec();
    }
}

interface ShenyuRequestCodec {
    byte[] encode(shenyu.Request request) throws Exception;
}

interface ShenyuResponseCodec {
    shenyu.Response decode(byte[] data) throws Exception;
}
```

This code defines a Spring Boot configuration class (`ShenyuKitexAutoConfiguration`) for integrating Kitex services with Apache ShenYu.

Key components:

- **Beans:**
  - `shenyuKitexPlugin`: Bean that creates a `ShenyuKitexPlugin` instance.
  - `shenyuKitexClient`: Bean that creates a `ShenyuKitexClient` instance.
  - (Optional) `shenyuRequestCodec`, `shenyuResponseCodec`: Beans for custom request/response serialization (if needed).

Functionalities:

- **Bean creation methods:**
    - `shenyuKitexPlugin`: Injects a `ShenyuKitexClient` and a list of `ShenyuMatcher` objects to create the plugin.
    - `shenyuKitexClient`: Injects required dependencies (`ShenyuClient`, `KitexCodec`) and optional codec beans to create the client. Performs null checks for critical properties.
    - (Optional) `shenyuRequestCodec`, `shenyuResponseCodec`: They provide custom logic for serializing requests and deserializing responses to/from ShenYu format (if needed).

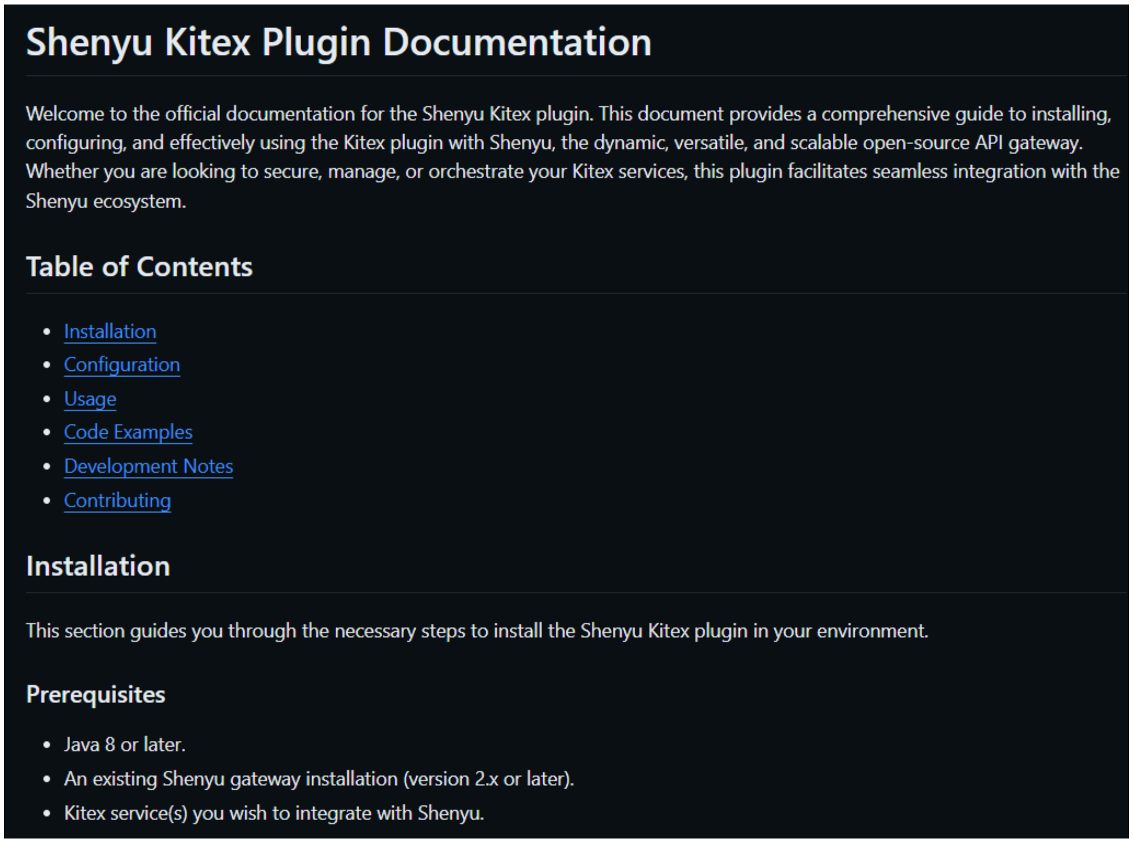
Overall, this configuration class simplifies the integration of Kitex services with ShenYu by automatically creating and injecting the necessary components.

Week 7 (July 8 - 14) - Midterm Evaluation:

- Prepare for the midterm evaluation by summarizing my progress, achievements, and any challenges encountered.
- Address any feedback from my mentors and refine the development plan for the remaining weeks.

Week 8-9 (July 15 - July 28):

- **Focus:** Documentation and Refinement
    - Finalize the Shenyu Kitex plugin documentation for the `shenyu-website` (https://shenyu.apache.org/), including usage instructions, configuration details, and code examples.
    - Conduct code reviews with my mentors and address any suggestions for improvement.
    - Refine unit and integration tests to ensure comprehensive coverage.
    - **Deliverable:** Comprehensive Shenyu Kitex plugin documentation and refined codebase with improved test coverage.



Shenyu Kitex Plugin Documentation

Welcome to the official documentation for the Shenyu Kitex plugin. This document provides a comprehensive guide to installing, configuring, and effectively using the Kitex plugin with Shenyu, the dynamic, versatile, and scalable open-source API gateway. Whether you are looking to secure, manage, or orchestrate your Kitex services, this plugin facilitates seamless integration with the Shenyu ecosystem.

Table of Contents

- Installation
- Configuration
- Usage
- Code Examples
- Development Notes
- Contributing

Installation

This section guides you through the necessary steps to install the Shenyu Kitex plugin in your environment.

Prerequisites

- Java 8 or later.
- An existing Shenyu gateway installation (version 2.x or later).
- Kitex service(s) you wish to integrate with Shenyu.

Note: *This document is a draft and subject to revision based on mentor feedback.*

Week 10-11 (July 29 - August 11):

- **Focus:** Final Deliverables and Testing
  - Conduct performance testing to evaluate the plugin's efficiency under load.
  - Address any final bugs or issues reported during testing.
  - **Deliverable:** Performance test results, and a bug-free Shenyu Kitex plugin.

Week 12 (August 12 - 18):

- **Focus:** Final Submission and Evaluation
  - Submit the final Shenyu Kitex plugin codebase and documentation as per GSoC guidelines.
  - Participate in the final mentor evaluation, addressing their feedback and questions.

Important Checkpoints(Coding period):

- **Week 1-2:** Functional `shenyu-client-kitex` module with unit tests.
- **Week 3-4:** Functional `shenyu-plugin-kitex` module with unit tests, integrated with `shenyu-client-kitex`.
- **Week 5-6:** Functional `shenyu-spring-boot-starter-plugin-kitex` and `shenyu-integrated-test-kitex` modules with integration tests.
- **Week 7:** Midterm evaluation with progress report and development plan refinement.
- **Week 8-9:** Comprehensive Shenyu Kitex plugin documentation and refined codebase with improved test coverage.
- **Week 10-11:** Performance test results, and a bug-free Shenyu Kitex plugin.
- **Week 12:** Final submission of codebase and documentation.

Other commitments

I'm available throughout this summer with no prior commitments.I am prepared to dedicate my full attention and time to ensuring the project's success. Rest assured, with no conflicting engagements, I will prioritize the project and work diligently to achieve its objectives within the designated time frame.

Results for the Apache Community

Key Benefits of Shenyu Kitex Plugin for Developers:

- **Technical Flexibility:** Leverage Kitex RPC alongside Shenyu's gateway and rule engine for microservice development within the Apache ecosystem.

- **Simplified Integration:** Pre-configured Spring Boot starter module streamlines integration for Spring Boot and Kitex users, reducing development time.
- **Enhanced Developer Experience:** Comprehensive documentation and test suite ensure a smooth development experience with clear usage instructions, configuration guides, and best practices.

Lasting Impact on the Apache Community:

- **Well-integrated Solution:** Establishes a well-documented foundation for using Shenyu with Kitex, providing long-term value.
- **Knowledge Base Expansion:** Contributes to the overall Shenyu knowledge base, benefiting developers and fostering a more informed community.
- **Community Growth:** Encourages collaboration and knowledge sharing within the Apache community, leading to further innovation in microservices.

In conclusion,this project empowers ShenYu users and simplifies development, leaving a lasting impact on flexibility, efficiency, and Apache knowledge.