



Ballerina

Modern Software Development Practices

September 2024

Hello!

Asma Jabir

asmaj@wso2.com | Technical Lead | [@ballerinalang](#) | **WSO2**

Dilhasha Nazeer

dilhasha@wso2.com | Associate Technical Lead | [@ballerinalang](#) | **WSO2**

About this Session

Coming Up

Introduction to Software Development

Fundamentals of API

API driven Development

Hands-On Session

Prerequisites

Ballerina

VSCoDe

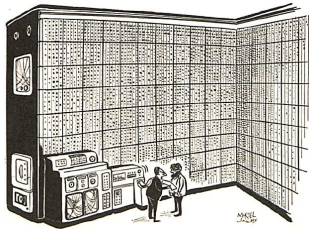
VSCoDe extension for Ballerina

Introduction to Software Development

Evolution of Software Development

Machine Level Programming

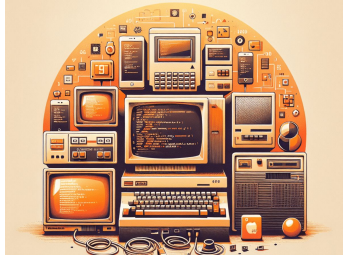
(1940s - 1960s)



Source:
<https://www.saturdayeveningpost.com/2023/06/cartoons-ok-computer/>

Structured Programming

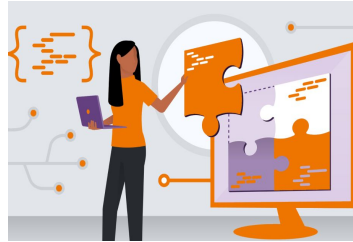
(1970s - 1980s)



Source:
<https://www.unimedia.tech/reviving-old-programming-languages-modern-software-development/>

Object Oriented Programming

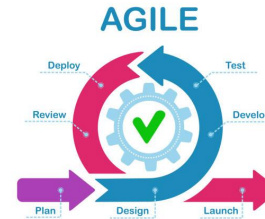
(1980s - 1990s)



Source:
<https://www.linkedin.com/pulse/using-design-patterns-oop-improve-code-structure-amr-saafan/>

Agile & Iterative Development

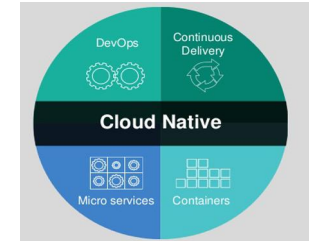
(2000s - 2010s)



Source:
<https://www.istockphoto.com/illustration/agile-iteration>

Automation & Cloud Native Development

(2010s - Present)



Source: digitalmediaworld.tv

Agile Development Practices

- Iterative Development
- Collaboration
- Continuous Feedback
- Customer-Centric

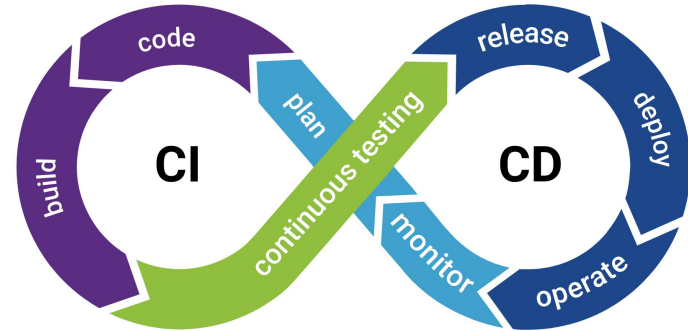


Source :

<https://targettrend.com/agile-methodology-meaning-advantagesdisadvantages-more>

DevOps Practices

- Continuous Integration (CI)
- Continuous Delivery (CD)
- Infrastructure as Code (IaC)
- Monitoring and Feedback



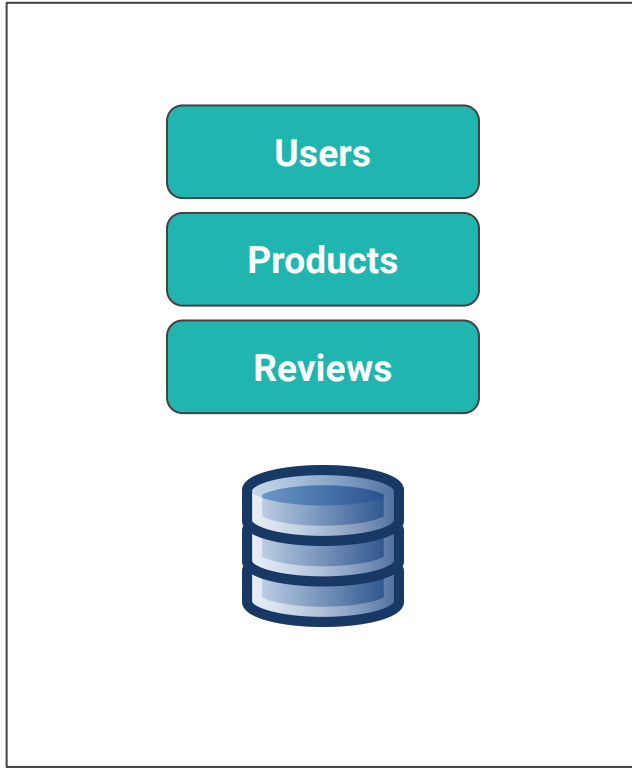
Source : <https://www.opsmx.com/blog/what-is-a-ci-cd-pipeline/>

Microservices

Microservice architecture is an architectural style that structures an application as a collection of services that are:

- Independently deployable
- Loosely coupled
- Organized around business capabilities
- Owned by a small team

Monolithic vs. Microservice Architecture



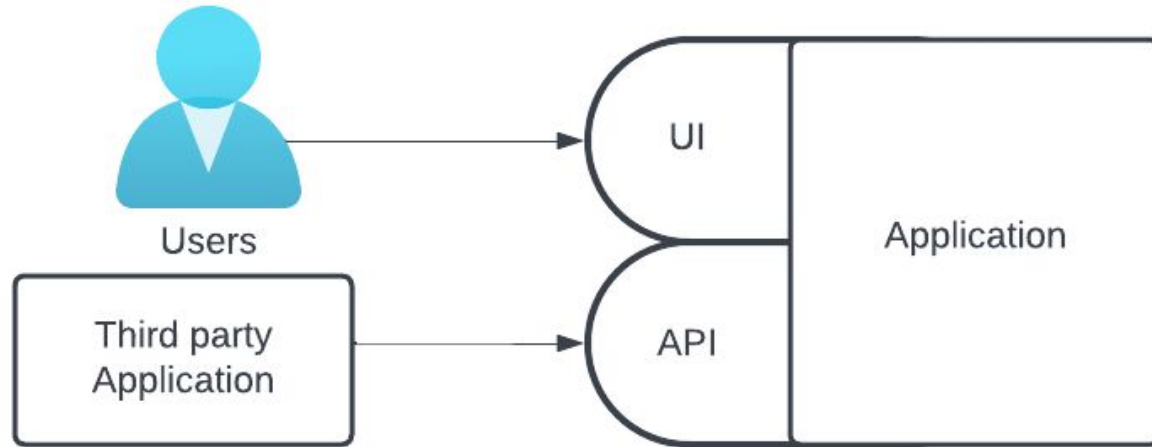
Monolithic Architecture



Microservice Architecture

Fundamentals of API

What is an API?



How does an API work?

- **Client and Server**

Client: The application or user making the request.

Server: The system/service that provides the data or functionality.

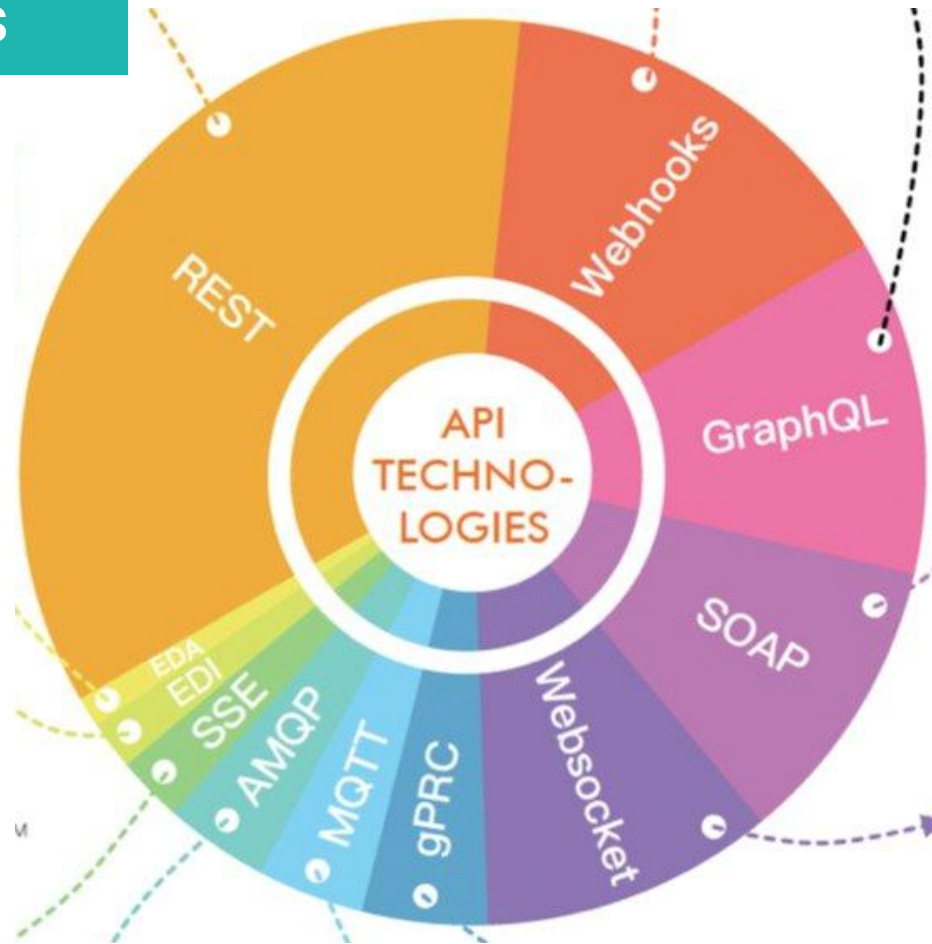
- **Requests and Responses**

- Common request methods: GET, POST, PUT, DELETE
- Response: Includes the requested data
or confirmation that an action was performed.

How does an API work?

- **Endpoints** : URLs where the client can make requests
- **Data Format** : Data is exchanged mostly in JSON or XML
- **Authentication**: Client specific 'API Key'
- **Rate Limiting**: to avoid overloading the server
- **Error Handling** : Common codes like 404, 400 and 500 are used

API Protocols



REST (REpresentational State Transfer)

- Most widely used architectural style leverages HTTP protocol
- Uses the concept of resources
- Resources can be accessed via verbs and resource paths
- Each resource has a standard format to represent data; server sends - client understands

GraphQL

- Relatively new protocol developed by Facebook
- Fast adaptation from the major companies
- Query language for APIs
- Data is structured as a hierarchical structure
- Has a single endpoint
- Clients can request exactly what they want, server responds with exactly what was requested

API-Driven Development

APIs are designed and developed first before building the application

Key Benefits:

- Modular Architecture
- Faster Development
- Scalability
- Improved Collaboration



Frontend Development

- **Languages & Technologies**

HTML

CSS

JavaScript

Frameworks & Libraries: React.js, Angular

- **Responsive Design**

Ensuring the website looks good and functions well on all devices

Frontend Development

- **UI/UX (User Interface/User Experience)**

Intuitive and accessible user interface

- **Performance Optimization**

Minimizing assets, lazy loading

- **Testing**

Using Jest, Cypress like tools to verify the functionality

Backend Development

- **Programming Languages**

Go, Python, Java

Ballerina

Frameworks : Spring Boot, Nest.js

- **Databases**

SQL - relational databases like H2, MySQL, MSSQL, PostgreSQL

NoSQL - MongoDB, Redis

Backend Development

- **Server & hosting**

 - On premises servers

 - Cloud providers like AWS, Azure, GCP

 - Containerized environments like Docker, Kubernetes

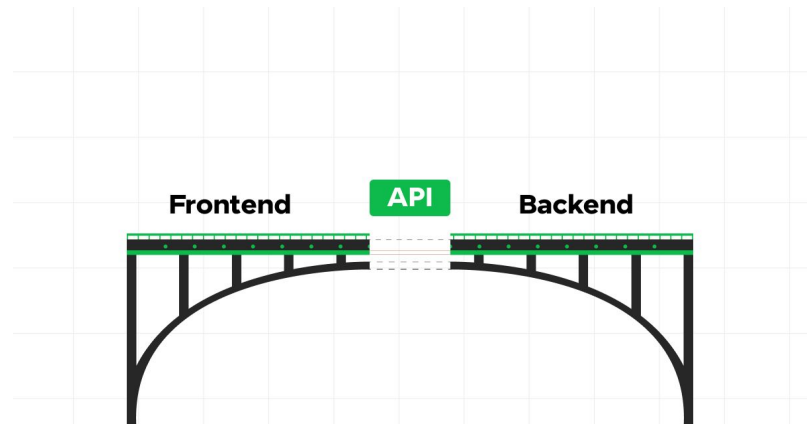
- **Observability & Monitoring**

 - Use of monitoring tools (e.g., Prometheus, Grafana)

 - Log management (e.g., ELK stack, Splunk)

Frontend and Backend Collaboration

- Clear Communication
- API Design
- Agile Workflows
- Agreements on Standards
- Testing
- Performance Optimization

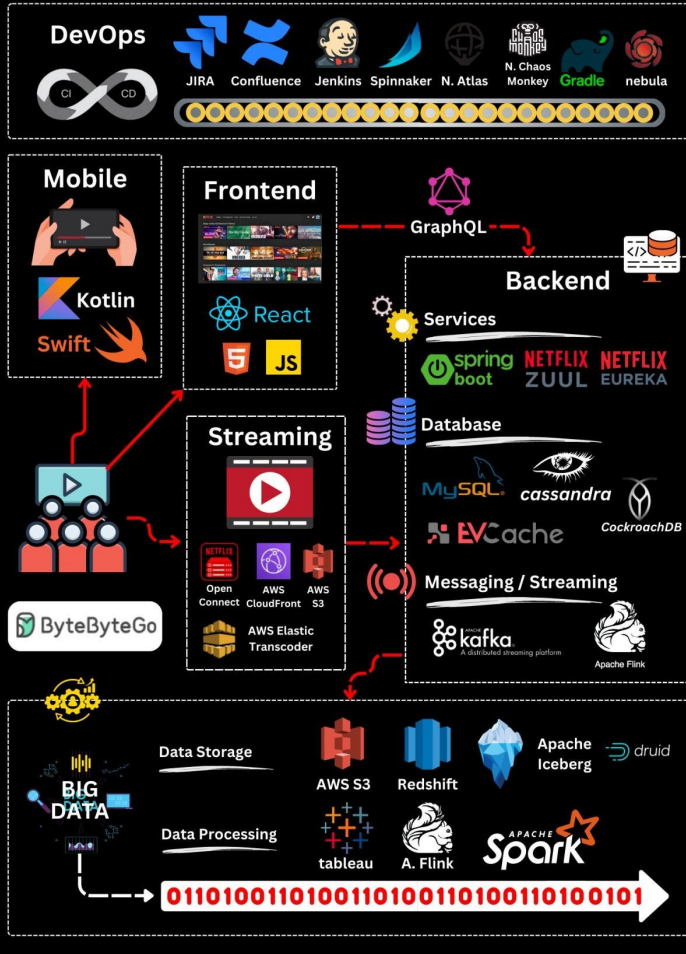


Real-world Examples

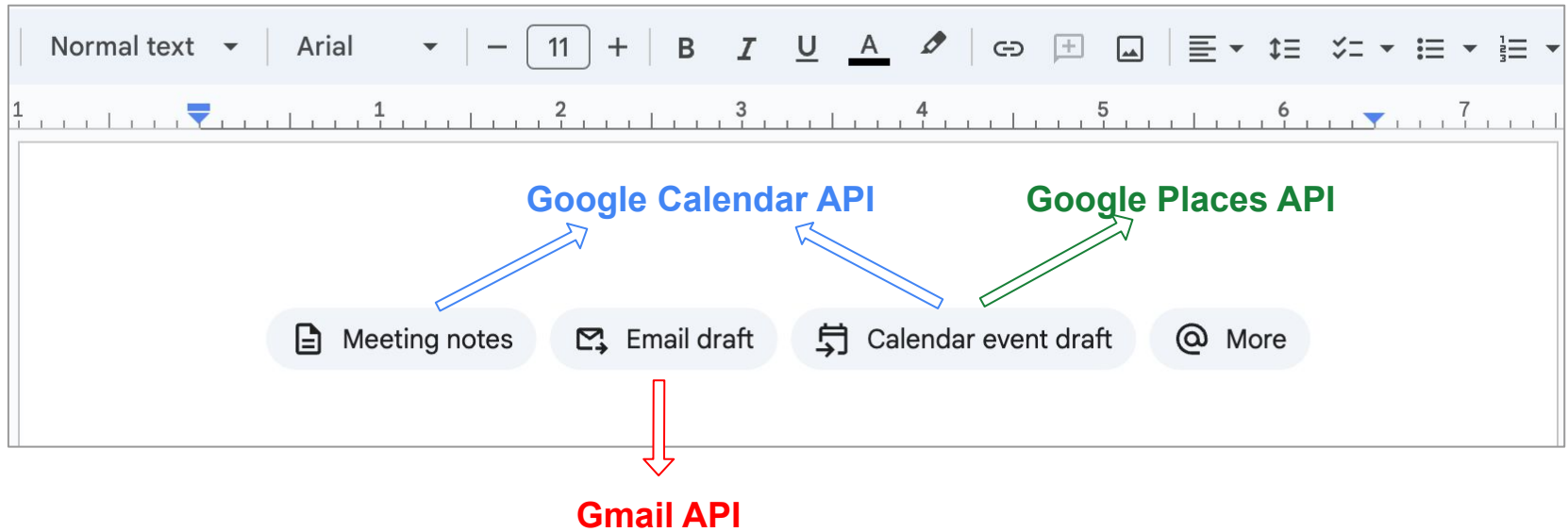
Sharing a post on Facebook



NETFLIX Tech Stack





Smartchips on GoogleDocs



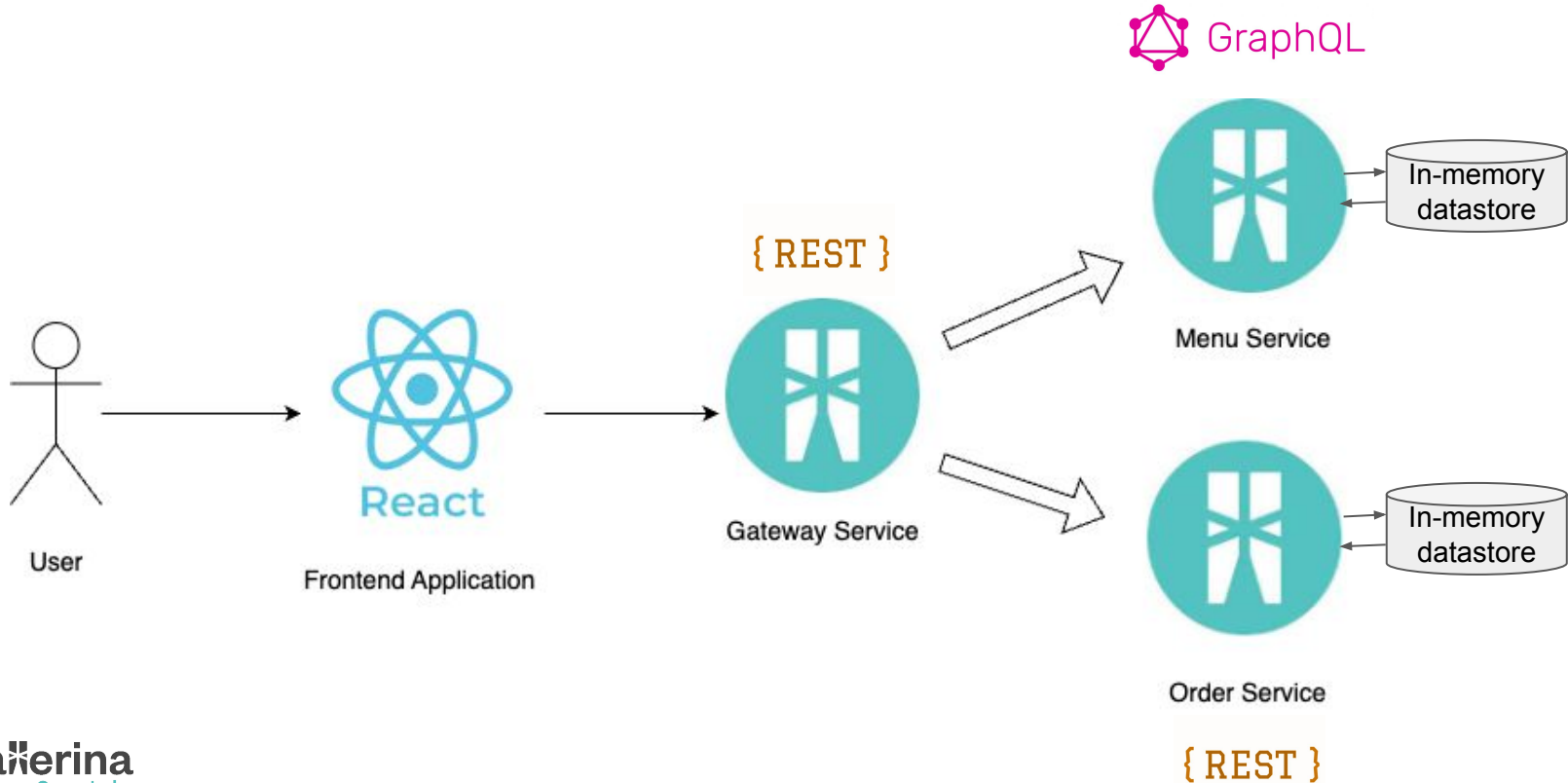
Hands-on

We will be building an order application...

- Based on Microservice Architecture
 - Menu Service 
 - Order Service 
- APIs to
 - Create new orders
 - View all orders
 - View details of a specific order
 - View all food items in the menu



Application Architecture



Ballerina Swan Lake

- Fully open-source programming language, powered by WSO2
- 6+ years of effort with 300+ contributors
- Cloud-native programming language optimized for integration
- Both textual syntax and graphical form
- Network Oriented Programming (DOP) paradigm

Ballerina is a full platform



Ballerina
SwanLake

Ballerina is a full platform

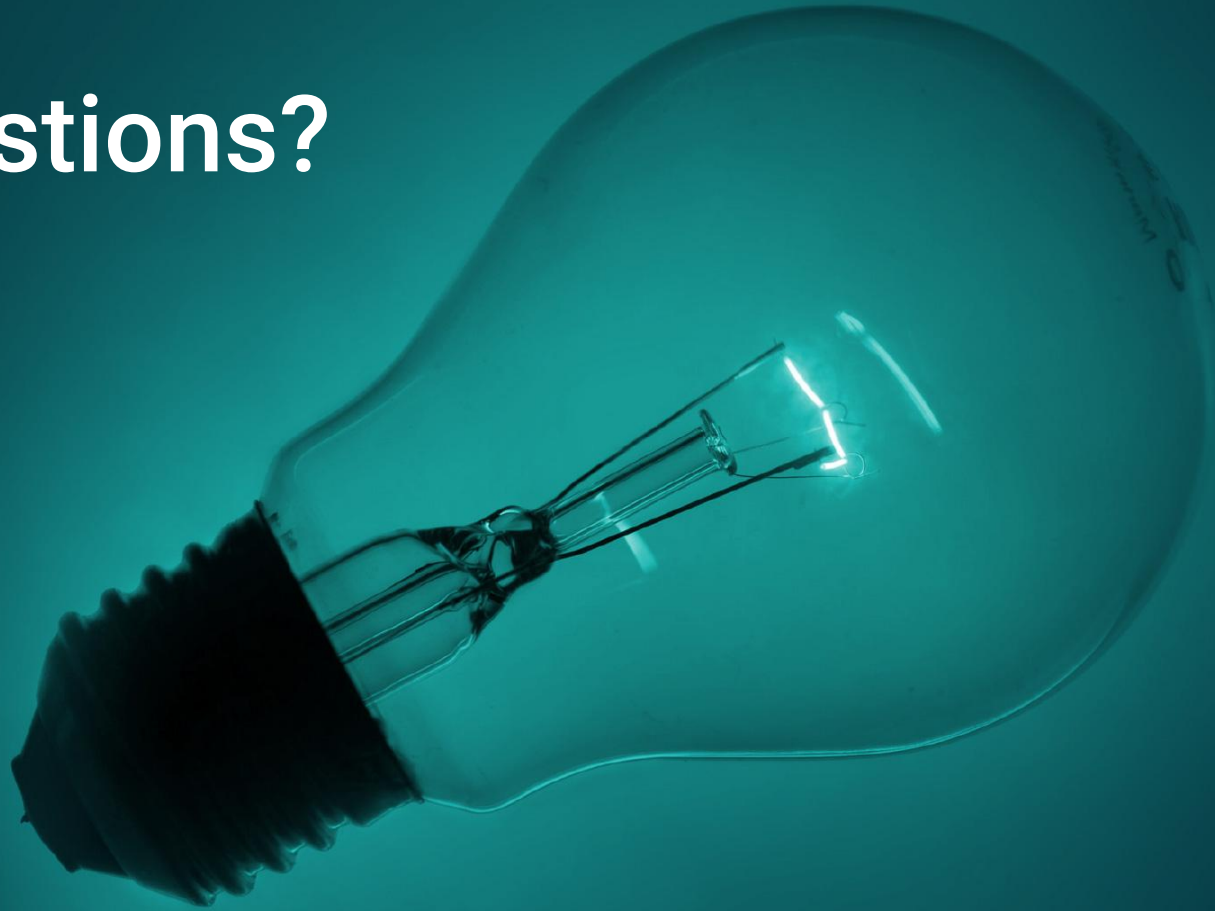
- VSCode plugin
 - Source and graphical editing
 - Debugging
- Tools for working with various protocols (REST, GraphQL, gRPC)
- Generate API documentation & test framework
- Ballerina standard library and extended library
- Ballerina Central (<https://central.ballerina.io/>)
 - Module sharing platform

Let's code

Source code:

https://github.com/azinneera/hotel_order_service

Questions?



Ballerina community

- Ballerina is an open source project
<https://github.com/ballerina-platform/ballerina-lang/>
- Seeking open source contributors
 - ◉ Ballerina is available for [hacktoberfest](#)
 - ◉ Has [good first issues](#) for external contributors
- Ballerina student engagement program
<https://ballerina.io/community/student-program/>



Find out more...

- Ballerina documentation
 - Ballerina use cases : Microservices
 - ballerina.io/usecases/microservices/
 - Ballerina by example
 - ballerina.io/learn/by-example/
 - API Documentation
 - <https://lib.ballerina.io/>
- Join the Ballerina community



Discord

[ballerinalang](https://discord.com/invite/ballerinalang)



COLLECTIVES[™]
on stackoverflow

[WSO2 Collective](https://www.stackoverflow.com/questions/tagged/wso2-collective)



[@ballerinalang](https://twitter.com/ballerinalang)



GitHub

[ballerina-lang](https://github.com/ballerina-lang)

Inter-university Ballerina Hackathon



The poster for the Ballerina Hackathon features a dark background with a spotlight effect. At the top left, the word "Ballerina" is written in white. To its right is a white icon of a person with a plus sign inside curly braces. Below this, the word "WIN" is written in large white letters, followed by a yellow sunburst icon. Underneath, the words "EXCITING PRIZES" are written in very large, bold white letters. A yellow banner below this text says "Registrations Are Now Open!". The main body of the poster shows a 3D podium with three steps. The top step is labeled "1st Place" and "Rs 150,000/-". The middle step is labeled "2nd Place" and "Rs 100,000/-". The bottom step is labeled "3rd Place" and "Rs 75,000/-". At the bottom right of the podium area, it says "Rs 10,000/- Each For 4th To 10th Places" and "Valuable Certificates For Every Submission". At the bottom left, there are logos for "University of Moratuwa IEEE Student Branch", "IEEE COMPUTER SOCIETY", and "WSO2".

 Register Now:
<https://lnkd.in/gmCdiBim>

Thank you!