Spring Boot с Amazon Web Services SDK: взаимодействие основных сервисов

Август Вилакия

Альфа-Банк

TWITTER @augustlakia TELEGRAM @augustlakia

Joker<?>

О спикере

- Больше 4-х лет опыта разработки на Java
- AWS Solution Architect
- Больше года опыта взаимодействия с инфраструктурой AWS

О чем будет доклад

О чем будет доклад

I. Перспективы использования AWS

II. Разбор типичной архитектуры в AWS

III. Spring Cloud AWS, как он помогает в разработке

IV. LocalStack, локальная разработка и тестирование

Зачем нам все это?


AWS
AWS EVERYWHERE
makeameme.org

Перспективы использования облачной инфраструктуры

- **Интуитивная настройка для высоконагруженных систем**

Перспективы знания облачной инфраструктуры

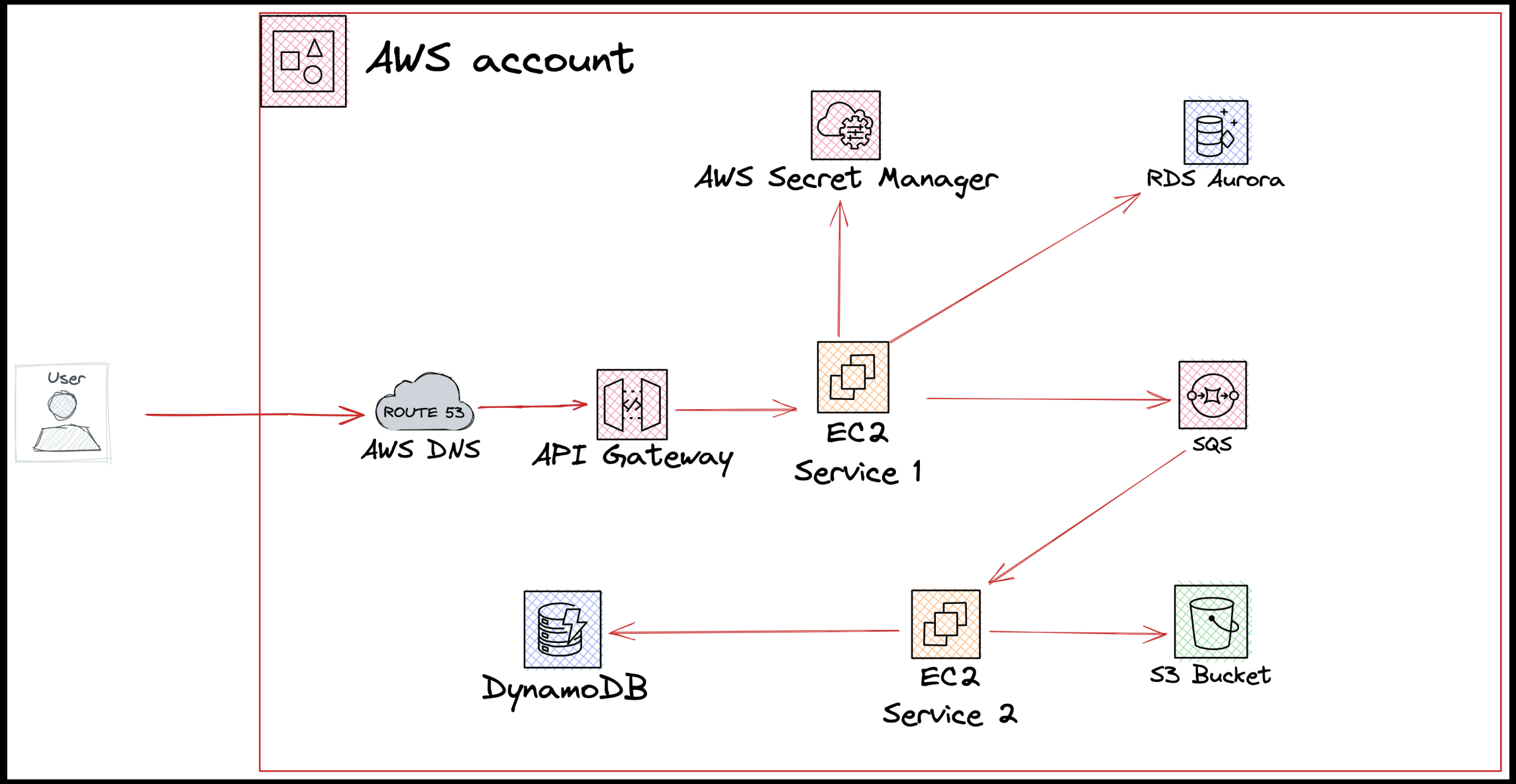
- Интуитивная настройка для высоконагруженных систем
- Санкции и сложность приобретения железа для своих серверов

Перспективы знания облачной инфраструктуры

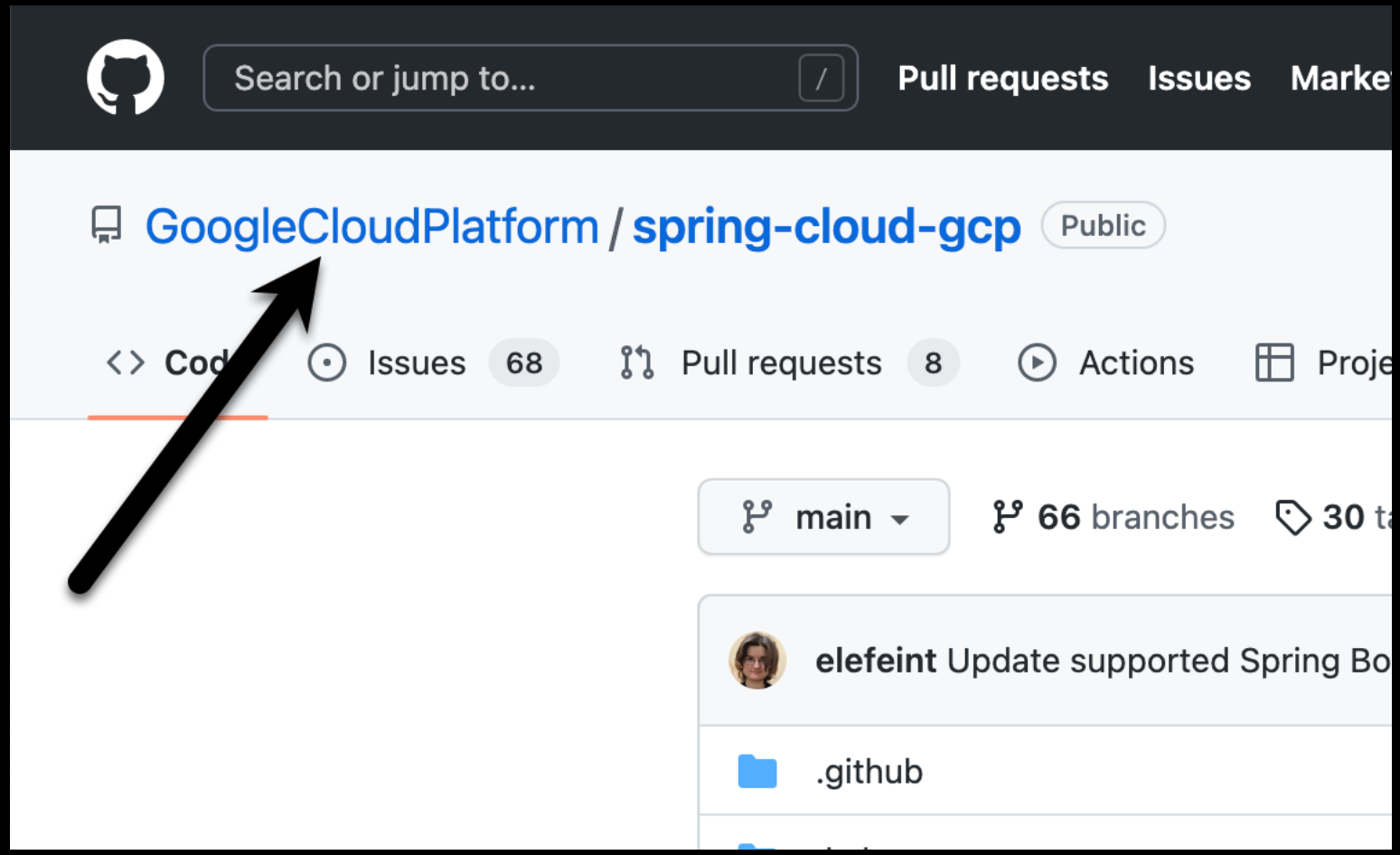
- Интуитивная настройка для высоконагруженных систем
- Санкции и сложность приобретения железа для своих серверов
- Больше 400-х упоминаний AWS в вакансиях Java разработчика на HeadHunter

Пример микросервисной архитектуры в AWS

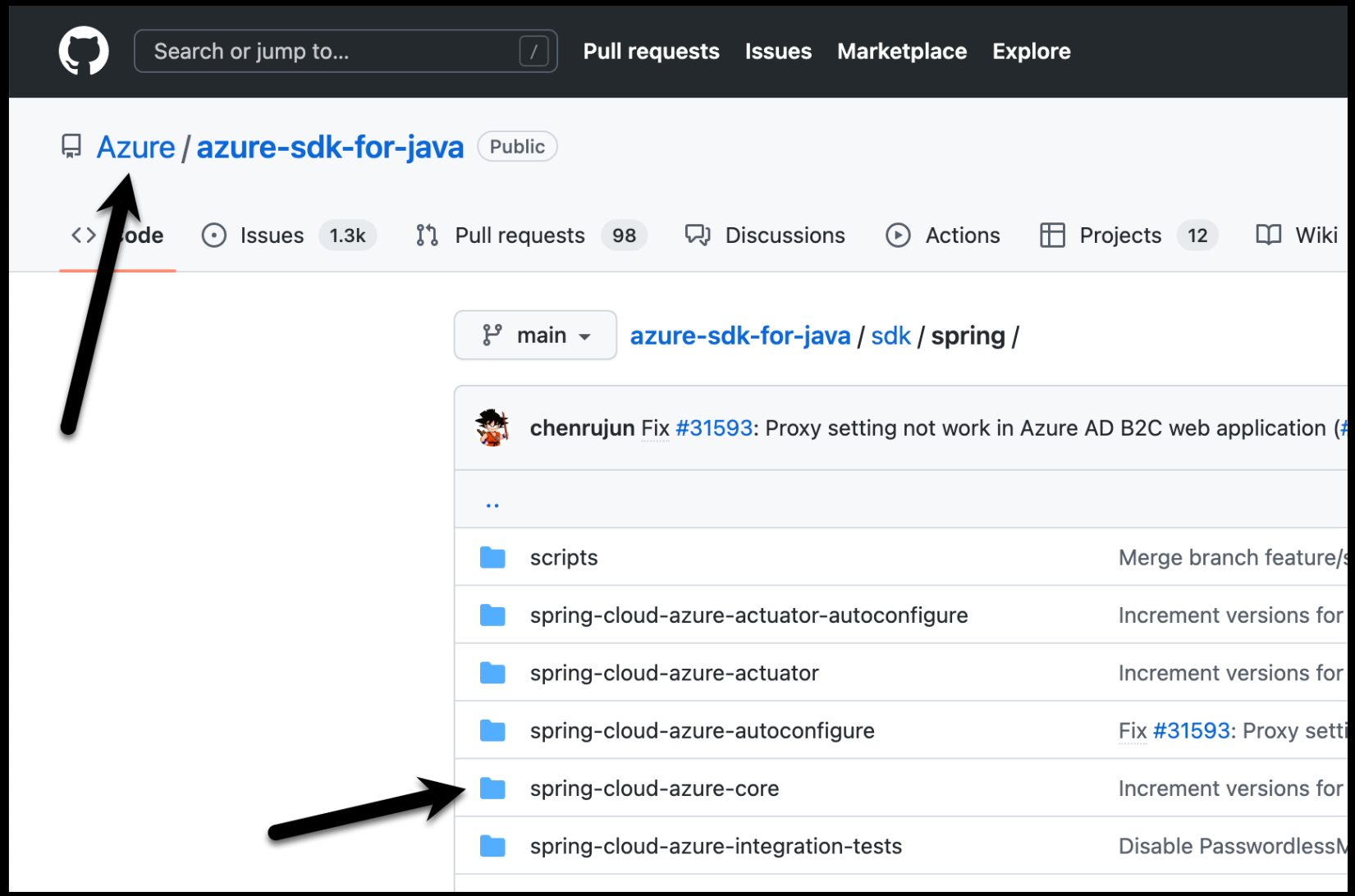

Is seventeen a lot?
Depends on the context!
Dollars? No.
Ways to deploy containers on AWS? Yes.
imgflip.com


AWS account
AWS Secret Manager
RDS Aurora
User
ROUTE 53
AWS DNS
API Gateway
EC2
Service 1
SQS
DynamoDB
EC2
Service 2
S3 Bucket

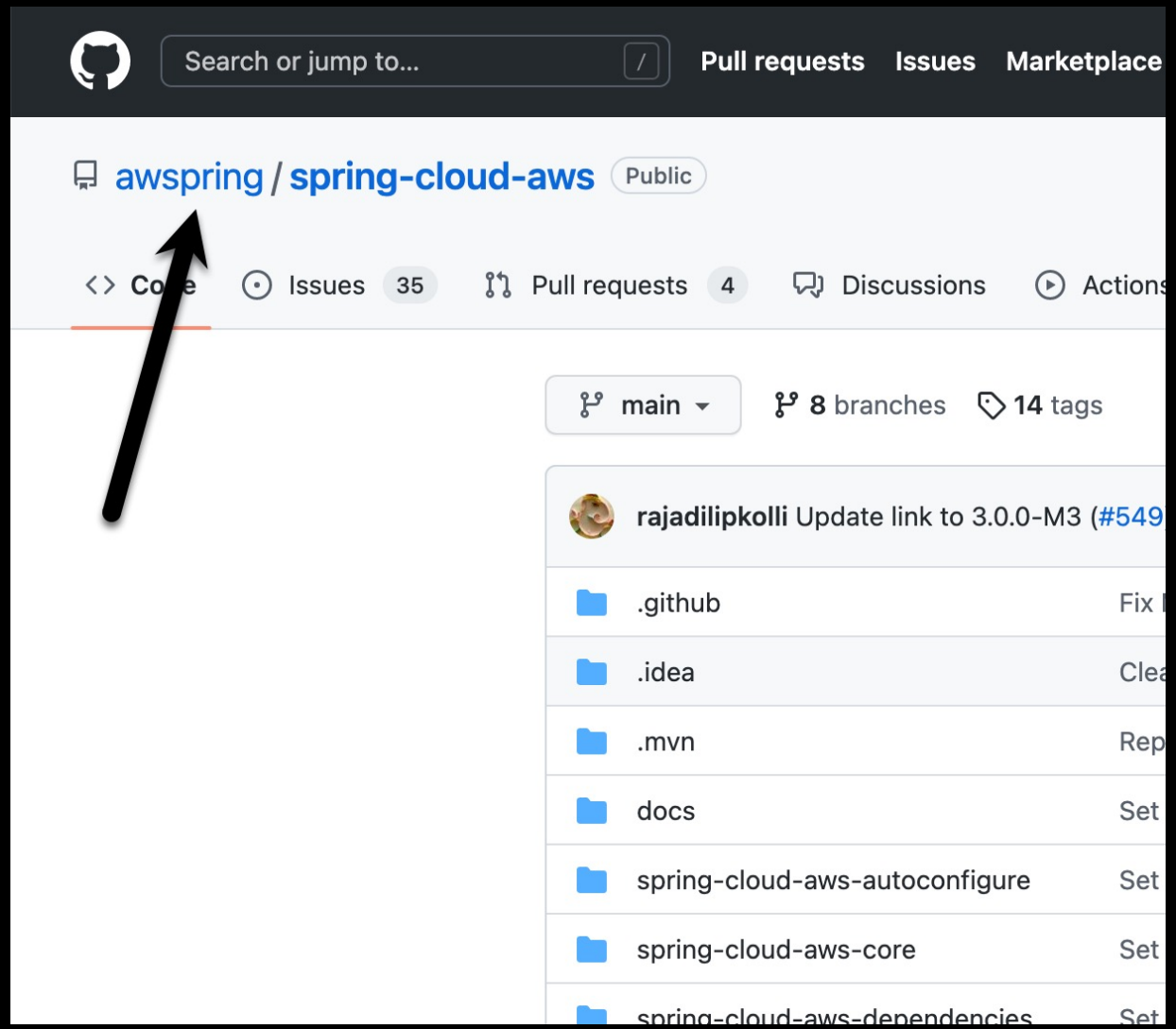
Spring Cloud GCP



Spring Cloud Azure



Spring Cloud AWS



Spring Cloud AWS

- Среди большой тройки облачных провайдеров, единственный community проект для взаимодействия с облаком. У Azure – проект поддерживается Microsoft, у GCP – Google.
- Проект помогает взаимодействовать с сервисами AWS для более удобной разработки.

Поддерживаемые сервисы

AWS Service	Spring Cloud AWS 2.x
S3	DONE
SNS	DONE
SES	DONE
Parameter Store	DONE
Secrets Manager	DONE
SQS	DONE
RDS	DONE
EC2	DONE
ElastiCache	DONE
CloudFormation	DONE
CloudWatch	DONE
Cognito	DONE
DynamoDB	NOT SUPPORTED

Поддерживаемые сервисы

AWS Service	Spring Cloud AWS 2.x	Spring Cloud AWS 3.x
S3	DONE	DONE
SNS	DONE	DONE
SES	DONE	DONE
Parameter Store	DONE	DONE
Secrets Manager	DONE	DONE
SQS	DONE	DONE
RDS	DONE	IN PROGRESS
EC2	DONE	NOT SUPPORTED
ElastiCache	DONE	NOT SUPPORTED
CloudFormation	DONE	NOT SUPPORTED
CloudWatch	DONE	DONE
Cognito	DONE	DONE
DynamoDB	NOT SUPPORTED	DONE

RDS Aurora vs EC2

RDS Aurora vs EC2

	RDS	EC2
Обновления	✅	🚫

RDS Aurora vs EC2

	RDS	EC2
Обновления	✅	🚫
Бэкапы	✅	🚫

RDS Aurora vs EC2

	RDS	EC2
Обновления	✅	🚫
Бэкапы	✅	🚫
Безопасность	✅	🚫

RDS Aurora vs EC2

Public accessibility Info

- ○ Yes
  EC2 instances and devices outside of the VPC hosting the DB instance will connect to the DB instances. You must also select one or more VPC security groups that specify which EC2 instances and devices can connect to the DB instance.
- ◉ No
  DB instance will not have a public IP address assigned. No EC2 instance or devices outside of the VPC will be able to connect.

RDS Aurora vs EC2

	RDS	EC2
Обновления	✅	🚫
Бэкапы	✅	🚫
Безопасность	✅	🚫
Настройка высоконагруженности и масштабируемости	✅	🚫

Подключаем RDS Aurora

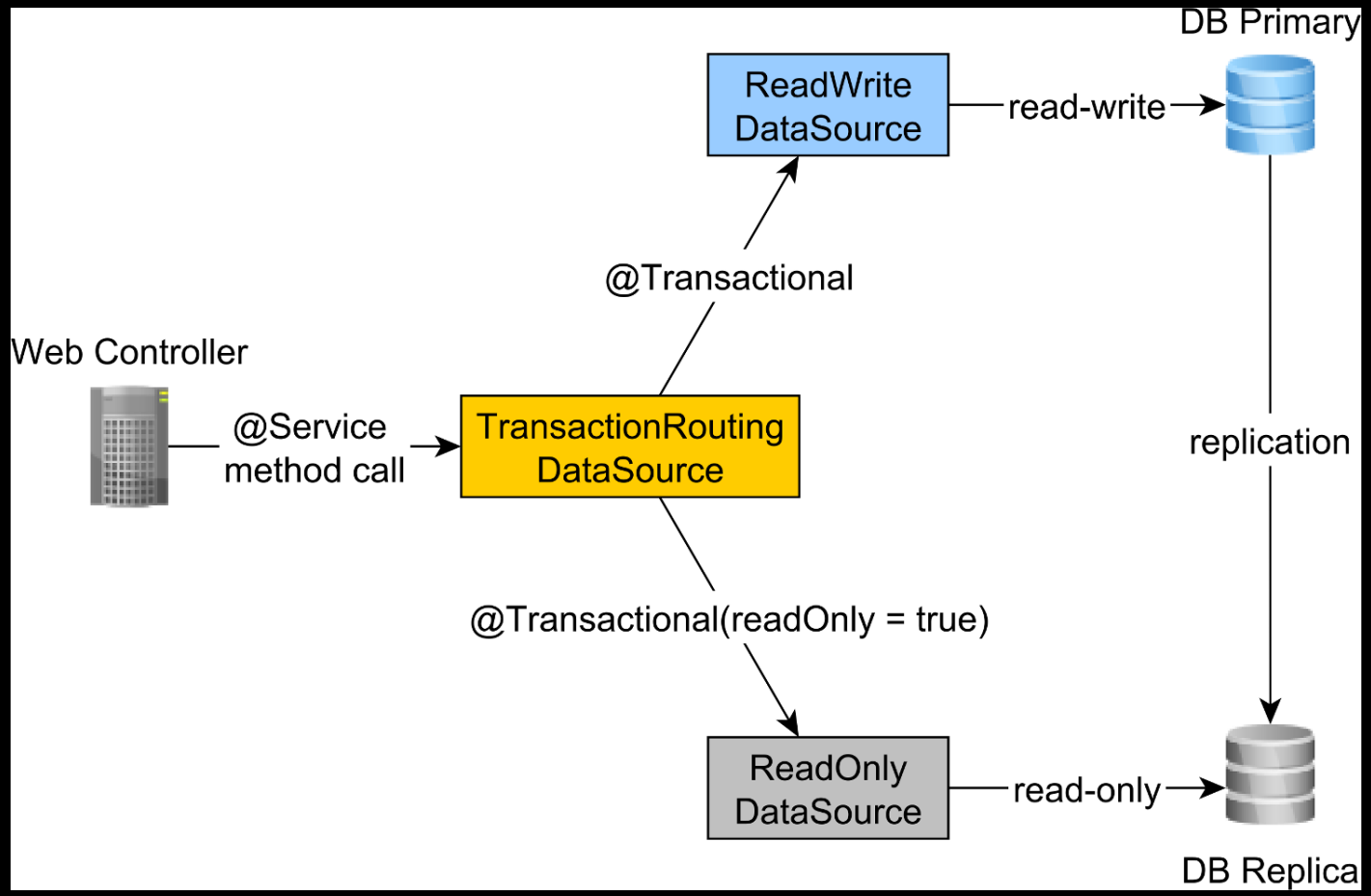
```yaml
cloud:
  aws:
    rds:
      instances:
        -
          db-instance-identifier: jokerconf
          database-name: jokeraurora
          username: joker
          password: joker
          read-replica-support: true
```

Использование read-replica

```
@Service
class UserService {
    @Transactional
    void registerUser(User user) {
        ...
    }

    @Transactional(readOnly=true)
    User findUser(Long id) {
        ...
    }
}
```

Как работает read replica



Подключаем Secret Manager

I. В данном случае при отсутствии секрета, ошибка

```
spring:
  config:
    import: aws-secretsmanager:/secrets/jokerconf
```

II. В данном случае ошибки при загрузке не будет

```
spring:
  config:
    import: optional:aws-secretsmanager:/secrets/jokerconf
```

Используем Secret Manager

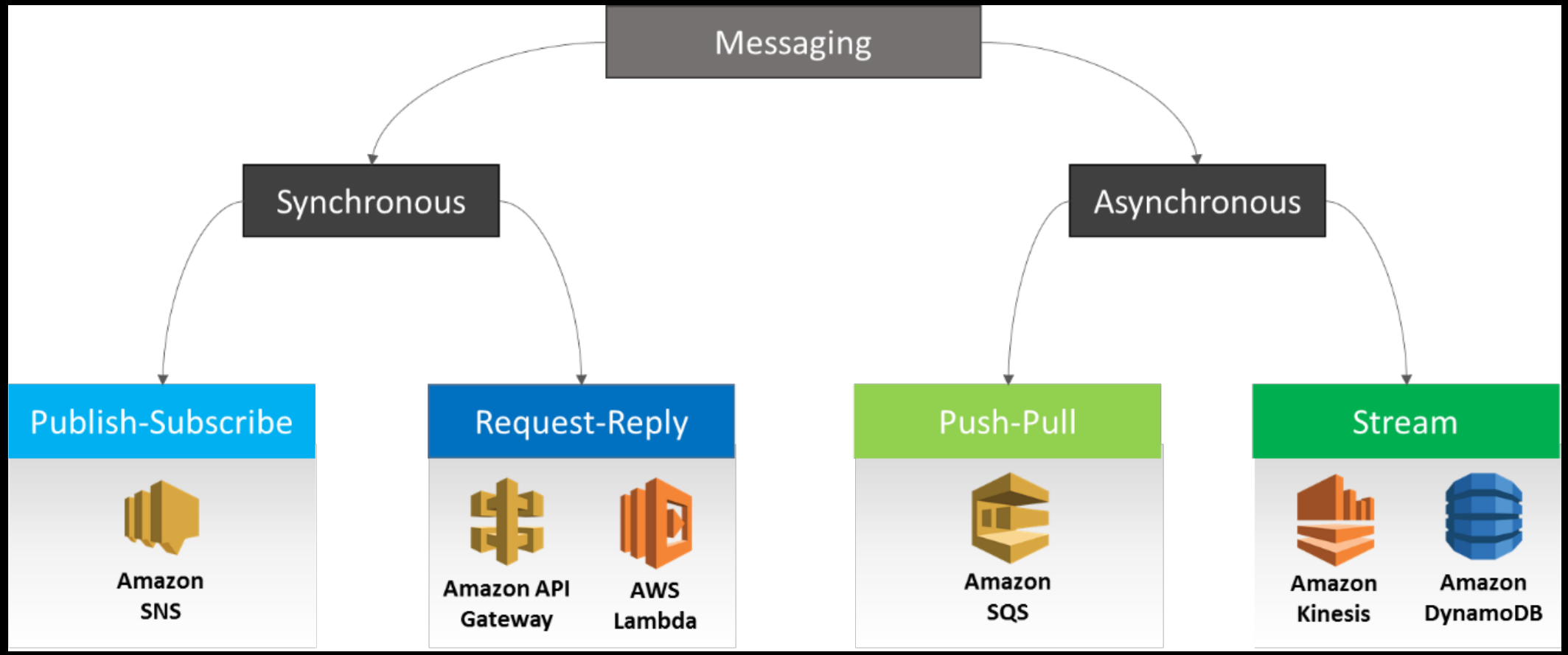
```
@Value("${username}")
private String username;

@Value("${password}")
private String password;
```

Используем Secret Manager

```yaml
spring:
  datasource:
    url: ${jdbc-url}
```

Брокеры сообщений, AWS



Simple Queue Service

- **Pull based брокер, т.е. консьюмеру необходимо самому прочитать сообщение**

Simple Queue Service

- Pull based брокер, т.е. консьюмеру необходимо самому прочитать сообщение
- Важное преимущество, это managed сервис, у вас нет ничего, все проблемы решаются с UI

Simple Queue Service

- Pull based брокер, т.е. консьюмеру необходимо самому прочитать сообщение
- Важное преимущество, это managed сервис, у вас нет ничего, все проблемы решаются с UI
- Старается сохранять очередность, но не всегда получается

Простое чтение из SQS

```
@SqsListener("JokerConf")
public void listen(String message) {
    System.out.println(message);
}
```

Параметры чтения

I. POJO – сам получаемый объект

II. List<Pojo>- включается батч обработка

III. @Headers – мапа с хедерами сообщения

IV. Acknowledgement – дает возможность ручного уведомления о получении

V. Visibility – возможность сообщение стоит ли показывать сообщения другим кто прочитает

Попробуем что-нибудь отправить

```
@Autowired
public QueueMessagingTemplate queueTemplate;

public void send(Human human) {
    queueTemplate.convertAndSend(
            destinationName: "QueueName", human);
}
```

S3 – Simple Storage Service

- Сервис AWS, в который можно загрузить объект/массив байт, получить ID и быть уверенным что он там останется
- На стороне клиента хранится база данных с метаданными файлов загруженными и при необходимости выгружается из AWS по ID
- Хранить терабайты и получать их каждый месяц, будет стоить пару тысяч рублей

S3 – все просто

```
@Autowired
public S3Client s3Client;

void readFile(String fileName, String bucketName) throws IOException {

    ResponseInputStream<GetObjectResponse> response =
            s3Client.getObject(
            request -> request
                    .bucket(bucketName)
                    .key(fileName));
```

Имя Бакета

Имя файла

S3Template

```
@Autowired
public S3Template s3Template;

void saveAndReadFile(String fileName, String bucketName) throws IOException {

    Human human = new Human("Muhammad", "Ali");

    s3Template.store(bucketName, fileName, human);

    human = s3Template.read(bucketName, fileName, Human.class);
```

Загружаем

Выгружаем

DynamoDB

- NoSQL, Schemaless, Key-Value, Document
- Автоматическая репликация между регионами во всем мире, поддержка глобальных и дополнительных индексов
- Есть два стартера, один который дает возможность использовать JPA, а другой как раз из SPRING CLOUD AWS

DynamoDB

```
@DynamoDbBean
public class Human {
    private UUID id;
    private String name;
    private String lastName;

    @DynamoDbPartitionKey
    public UUID getId() {
        return id;
    }

}
```

DynamoDB

- **DynamoDbTemplate – бин, который дает нам доступ к CRUD**

```
dynamoDbTemplate.save(human);
dynamoDbTemplate.update(human);
dynamoDbTemplate
        .delete(Key
        .builder()
        .partitionValue("SOME KEY"));
```

LocalStack

- Встает вопрос, как все это тестировать? Есть несколько вариантов
- Есть вариант лучше, LocalStack – локальная эмуляция сервисов AWS
- Как можно было понять, я очень не люблю писать команды в консоли. Тут есть user-friendly UI

LocalStack установка

```
➜  ~
➜  ~ python3 -m pip install localstack
```

```
➜  ~
➜  ~ export LOCALSTACK_API_KEY=TEST_API_KEY
```

LocalStack доступные сервисы

Resource Browsers

App Integration

- AppSync
- API Gateway
- SQS
- SNS
- Step Functions
- EventBridge

Compute

- EC2
- Lambda
- ECS

Management/Governance

- CloudFormation
- CloudWatch Logs
- CloudWatch Metrics
- Systems Manager
- CloudTrail

Business Applications

- SES

Security Identity Compliance

- Cognito
- Secrets Manager
- KMS
- IAM

Storage

- S3
- Backup

Database

- Dynamo DB
- RDS
- ElastiCache
- Timestream DB

Analytics

- Athena
- Kinesis

LocalStack co Spring Cloud AWS

SQS

Name	Queue Url
jokerTest	http://localhost:4566/000000000000/jokerTest

LocalStack co Spring Cloud AWS

```yaml
spring:
  cloud:
    aws:
      s3:
        endpoint: http://localhost:4566
      sqs:
        endpoint: http://localhost:4566
      dynamodb:
        endpoint: http://localhost:4566
```

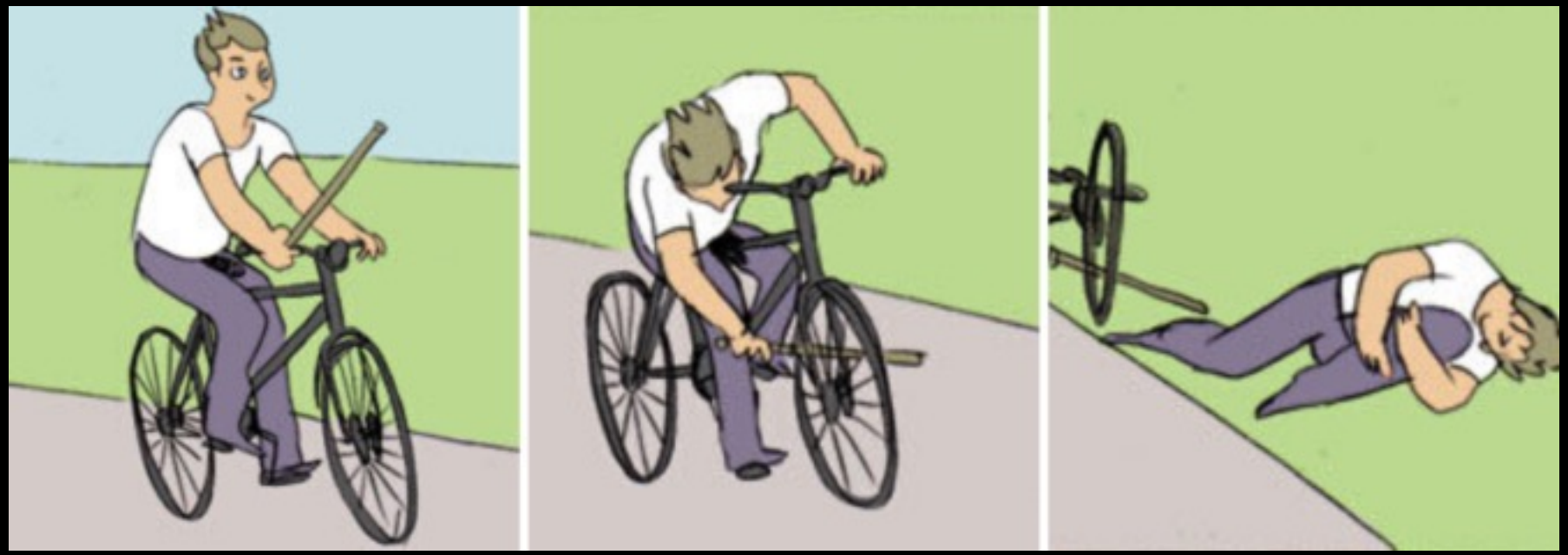
Интеграционное тестирование

```
@SpringBootTest
@Testcontainers
public class LocalStackTest {

    @Container
    static LocalStackContainer localStack = new LocalStackContainer(DockerImageName.parse("localstack/localstack"))
            .withServices(LocalStackContainer.Service.S3);

    @DynamicPropertySource
    static void localStackProperties(DynamicPropertyRegistry registry) {
        registry.add( name: "spring.cloud.s3.endpoint",
                () -> localStack.getEndpointOverride(LocalStackContainer.Service.S3));
    }
```

А что есть у нас?



Яндекс Облако

- Несколько десятков совместимых с Amazon API
- Можно использовать Amazon SDK для работы с очередями, YDB, S3

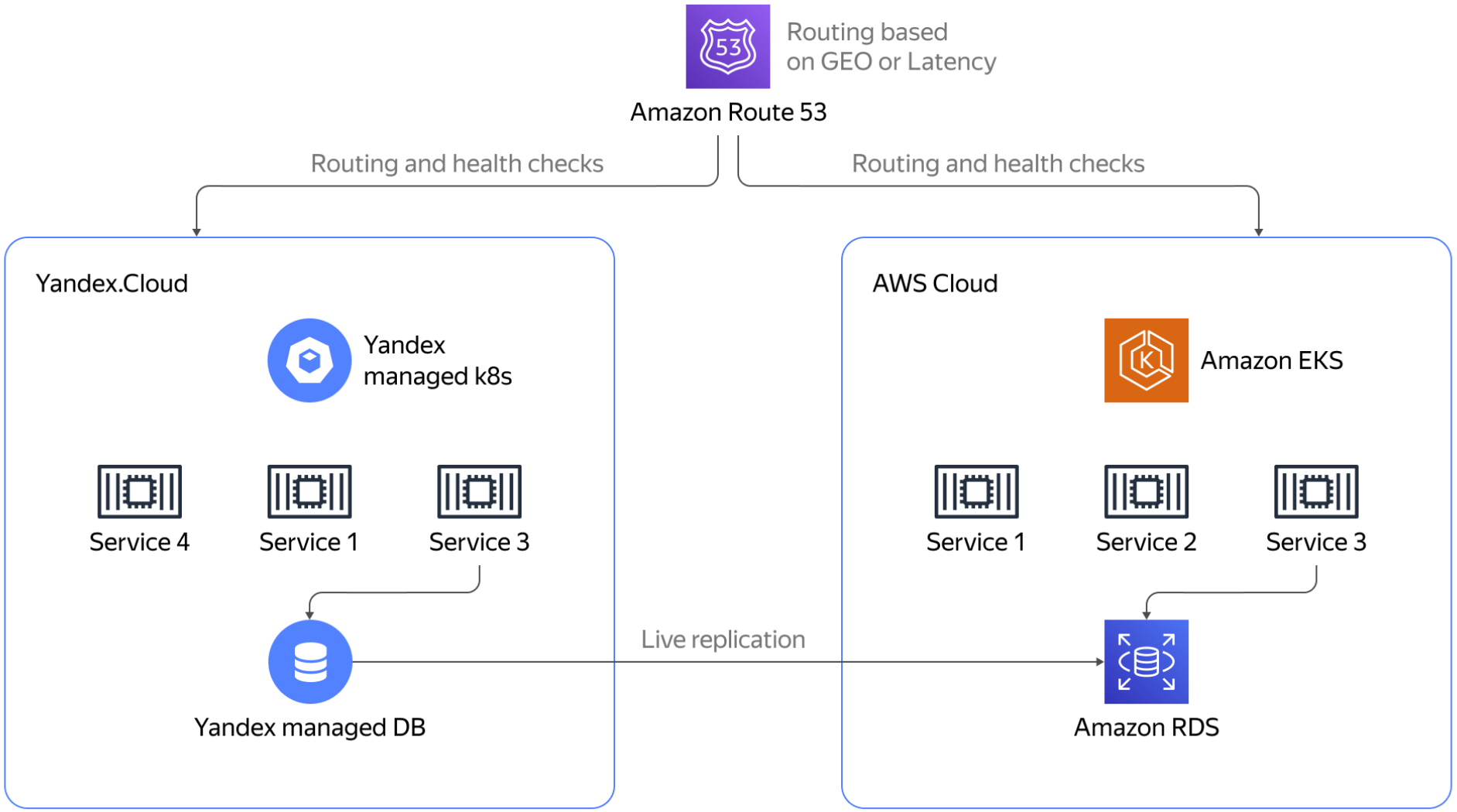
Совместимость с Amazon SDK наглядно

```
BasicAWSCredentials awsCreds = new BasicAWSCredentials( accessKey: "yandex_key_id",  secretKey: "yandex_secret");

AmazonS3 s3 = AmazonS3ClientBuilder.standard()
        .withCredentials(new AWSStaticCredentialsProvider(awsCreds))
        .withEndpointConfiguration(
                new AmazonS3ClientBuilder.EndpointConfiguration(
                        serviceEndpoint: "storage.yandexcloud.net", signingRegion: "ru-central1"
                )
        )
        .build();

s3.getObject( bucketName: "YandexBucketName",  key: "Yandexkey");
```

Использовать AWS для зарубежных пользователей, а для РФ использовать Я.Облако


Routing based
on GEO or Latency
Amazon Route 53
Routing and health checks
Routing and health checks
Yandex.Cloud
Yandex
managed k8s
Service 4
Service 1
Service 3
Yandex managed DB
Live replication
AWS Cloud
Amazon EKS
Service 1
Service 2
Service 3
Amazon RDS

Подводные камни использования managed сервисов

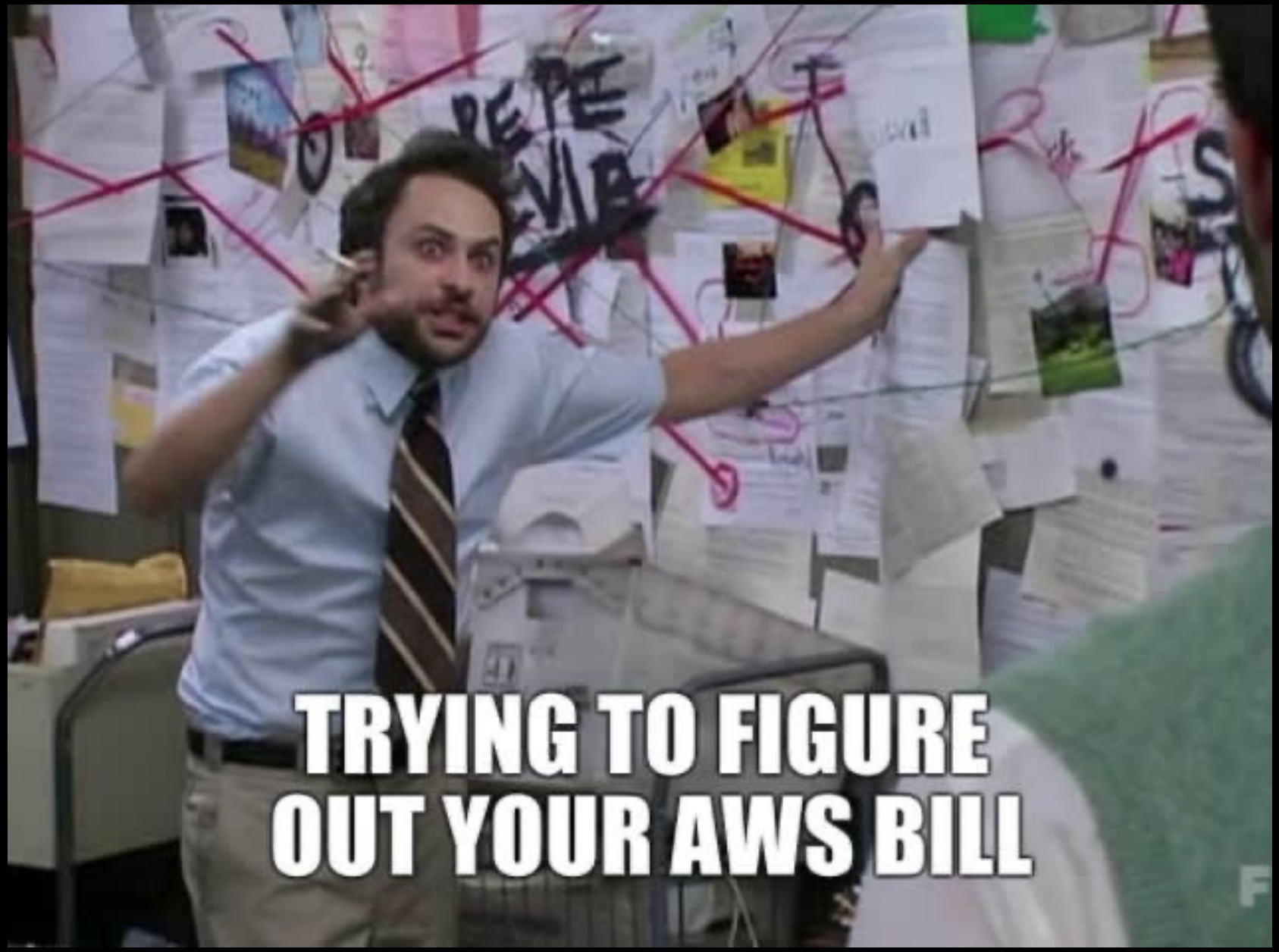
Вендор лок

Вендор лок

- Общепринятые решения с взаимозаменяемым АПИ

Вендор лок

- Общепринятые решения с взаимозаменяемым АПИ
- SOLID, буквы I и D что-то да означают


TRYING TO FIGURE
OUT YOUR AWS BILL

Дорого

Какой из всего вывод ?

Спасибо за внимание

<- Telegram