Java Code challenge -

Here is an attempt to implement FundTansferService.

Solution contains Service and Controller classes for Fund Transfer Service exposed as REST service.

The 'initiateTransfer' method of FundTransferService does the job of transferring funds from Source to target account.

FundTransferControllerTest and FundTransferServiceTest contain set of positive and negative test cases.

Deadlock - There are two ways to avoid dead lock for the said problem.

1. Define lock ordering based on some criteria to acquire lock on account objects, document it and enforce it in review process. Lock should be acquired on object in same order every time. Since there is no automated way to restrict developer to alter the lock ordering the approach is error prone.
2. More preferred approach is to proceed only if both locks are acquired else release both locks.

To make the application production ready we need to do - Publish rest service end points to service registry, Netflix Eureka can be used for this purpose. To externalize the configurations (log configuration in this case), Spring cloud config can be used.

Deployment –

War deployment Application can be as war in any servlet container Or Since it’s a Spring Boot application, can be run as standalone application in its embedded container.

Proposed enhancement –

Add API Gateway using Zuul proxy.

Add Circute Breaker to avoid cascading failure arises due to network latency and failures.

Add Load balancer for scalability Add centralize log analysis Add monitoring service