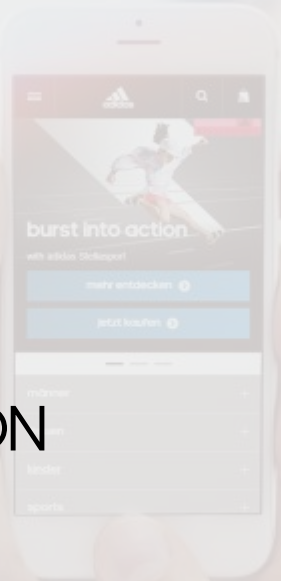
CODING CHALLENGE

WEB SERVICES AND INTEGRATION



Consumer engagement

#Digital IT | Consumer engagement

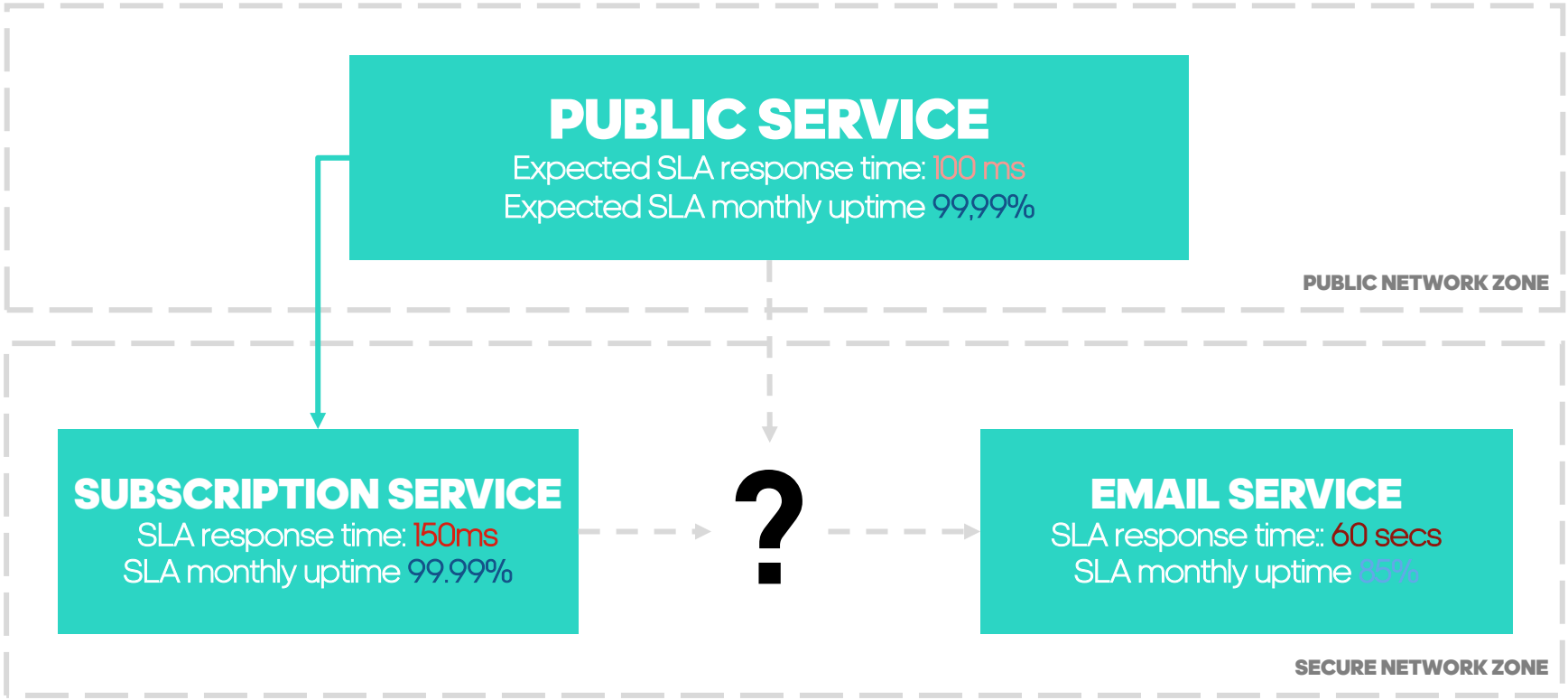
ARCHITECTURE PRINCIPLES

AT ADIDAS, WE CARE ABOUT SERVING CONSUMERS (IN THE SENSE OF APPLICATIONS MAKING USE OF OUR DATA AND SERVICES) BY SERVING MORE (HIGH VOLUME) , SERVING BETTER (LOW LATENCY) OR SERVING MORE RELIABLY(HIGH RESILIENCE). IN GENERAL, THE TOPIC OF HIGH AVAILABILITY IS VERY IMPORTANT TO BE READY FOR THE FUTURE AND DEAL WITH THE CURRENT STATE WITH CONFIDENCE.

IN A NUTSHELL, THE BASIC ARCHITECTURE PRINCIPLES ARE:

- ARCHITECT SOLUTIONS IN A WAY THAT MAKES THEM RE-USABLE
- COMPONENTS ARE DESIGNED TO SCALE BASED ON INCOMING LOAD.
- ALWAYS CARE ABOUT DATA & IT SECURITY
- ALL SOLUTIONS SHOULD TREAT CORE OPERATIONAL DATABASE AS A COSTLY RESOURCE THAT NEEDS TO BE USED ECONOMICALLY
- DESIGN FOR FAILURE – YOUR SOLUTION HAS TO ASSUME AND HANDLE EXCEPTION CONDITIONS

ARCHITECTURE LANDSCAPE AND REQUIREMENTS



MISSION STATEMENT

With the information given and additional assumptions of yours, you should develop an API for SUBSCRIPTIONs using the Java framework of your choice (ideally use Spring Boot).

To create subscriptions a single endpoint will be created: in the public service.

- Subscriptions contain: email, firstName, gender, dateOfBith, flag for consent and the newsletter Id corresponding to the campaign. Only gender and firstName are optional values. Rest of services receive the same parameters.
- Services must be secure, only public service is accessible from the end user.
- TIP: an example of a frontend application making use of this service would be similar to https://www.adidas.co.uk/on/demandware.store/Sites-adidas-GB-Site/en_GB/Newsletter-Subscribe

The subscription service has to persist the subscription and returns the ID of the created subscription to the public service.

To complete the subscription process, once the subscription was persisted by the subscription service, someway,, the email service will receive the required information to send an email to the user (take in account the SLAs to choose the best approach for this communication).

WHAT WE EXPECT FROM YOU

- Develop this application with a microservice approach,, all services must run independently from each other.
- Run each microservice in a different docker container linking them with any docker technology, no service discovery needed.
- Create a README.md explaining how to build/run/use the app, naming every framework/library you use and state what it does and why you used it.
- Design a solution that satisfy the SLAs..
- Every person having Java and some standard tools (Ant, Maven, Gradle) should be able to check out the code, build and run the app locally.
- Provide API documentation using swagger, API Blueprint or similar.
- Please use english as documentation language.
- BONUS 1: Create 1 slide with a CI/CD pipeline proposal for the app.
- BONUS 2: Design an architecture to deploy all services in a kubernetes cluster.
- When you're done check in your solution into any public GIT repo hoster (github, bitbucket, etc.) and send us the link and any other documentation you want.