Oregon State University
Ecampus

CS361: Assignment 9: Sprint 3 Plan (for Milestone #3)

Overview

Plan Microservices B, C, and D you'll make for your main program to call.

Instructions

Complete each item below by replacing the highlighted text (**Usability note**: double-click the text to select it).

1) What is your **Sprint Goal**? (e.g., fully implement the spell-checker, grammar checker, email address validation microservices) The Sprint Goal must clearly communicate what each of the B, C, and D microservices will do.

The Sprint goal for Sprint 3 is to create microservices that will be used in my main program, generate random list of variables. Such as measurements and integers.

2) Define **at least two user stories** for each microservice (B, C, and D). Provide your user stories and their functional and non-functional acceptance criteria (and associated quality attributes).

   Requirements for *each* microservice:

   - You must implement at least two user stories for each microservice.
   - Each user story must have a name.
   - Each user story must use the "As a… I want to... so that…" format.
   - Each user story must have at least one functional acceptance criterion.
   - All functional acceptance criteria must use the "Given… when… then…" format.

   Requirements for the *set of* microservices:

   - At least three different quality attributes must appear at least once on a user story's "back of index card".
   - Each quality attribute must be converted to a non-functional requirement.

Microservice B:

First user story

(Front of index card) List of Measurement Names As a user, I want to be able to start a random generator that will give me a randomized name from a list, so that it could be used as the first input for labeling my measurement before conversion in the main program.

(Back of index card)

Acceptance criteria

Functional requirements

- Given when you start the microservice by running it, it will generate a random measurement name that is in the given list.

Quality attributes & Non-functional requirements

- Functionality: Once the microservice is running, it will run and print the data collected in the program and write it in a text file.

Second user story

(Front of index card)

Random Generator

As a user, I want to be able to get a random Measurement name for a given list, so that I could use that data in the main program to continue the functionality.

(Back of index card)

Acceptance criteria

Functional requirements

- Given when you start the microservice, after receiving a request, the randomizer will start the generator to get a name from the list.

Quality attributes & Non-functional requirements

- Correctness: Once the microservice is active, it will make sure that the randomizer will give a name from the list, and not a random name.

Microservice C:

First user story

(Front of index card)

Number List

As a user, I want to be able to run a microservice that will have a list that will be used to generate a random number from.

(Back of index card)

Acceptance criteria

Functional requirements

- Given when the microservice is started, it will display a list from a set number to another number it will generate a random number from.

Quality attributes & Non-functional requirements

- Accuracy: Once the microservice is running, it will run according to how it should be, without making errors.

Second user story

(Front of index card)

Randomizer

As a user, I want to be able to run a randomizer, that will generate a random number to be used in the main program.

(Back of index card)

Acceptance criteria

Functional requirements

- Given when the microservice is running, it will read requirements from a text file that will activate a randomizer.

Quality attributes & Non-functional requirements

- Responsiveness: Once the microservice is active, it will give an immediate response to the user.

Microservice D:

First user story

(Front of index card)

Second List of Measurement Names

As a user, I want to be able to start a random generator that will give me a randomized name from a list, so that it could be used to convert the first input into the second input in the main program.

(Back of index card)

Acceptance criteria

Functional requirements

- Given when you start the microservice by running it, it will generate a random measurement name that is in a given list.

Quality attributes & Non-functional requirements

- Functionality: Once the microservice is running, it will run and print the data

collected in the program and write it in a text file.

Second user story

(Front of index card)

Randomizer

As a user, I want to be able to get a random Measurement name for a given list, so that I could use that data to convert the first input into the second input in the main program.

(Back of index card)

Acceptance criteria

Functional requirements

- Given when you start the microservice, after receiving a request, the randomizer will start the generator to get a name from the list.

Quality attributes & Non-functional requirements

- Correctness: Once the microservice is active, it will make sure that the randomizer will give a name from the list, and not a random name.

3) What kind of **communication pipe** will each microservice use? (e.g., text files, REST API)
Note: You can use the same type of communication pipe for all three microservices or different types.

Text files

This would be a good time to make a new repository for each of your microservices.

Submission

Upload a document in PDF or Word format via Canvas.

Grading

You are responsible for satisfying all criteria listed in the Canvas rubric for this assignment. You will be able to revise this assignment if you miss points.

Questions?

Please ask via Ed so that others can benefit from the answer.