UNIVERSITY OF OTAGO EXAMINATIONS 2019

Information Science

INFO 303

Enterprise Information Systems Infrastructure
Semester Two

(TIME ALLOWED: 3 HOURS)

This examination paper comprises 5 pages.

Candidates should answer questions as follows:

Answer ALL questions (total of 100 marks).

The following material is provided:

Nil

Use of calculators:

No calculators permitted.

Candidates are permitted copies of:

Nil

Other Instructions:

Nil

1. **[The suggested time allocation for answering this question is 30 minutes.]**

   Suppose you are responsible for designing a REST API to retrieve and update information about all papers taught at a university. Your API should provide the following functionality:

   i. *For a given department (identified by a four letter code such as `INFO`), return a collection representing all papers taught by that department.* Each item in the collection will be an appropriate RESTful identifier that can be used to retrieve information about a specific paper.

   ii. *For a given paper (identified by its RESTful identifier), return the paper details: paper code, title, description, points value and semester.* For simplicity, you can assume each paper is only offered in a single semester.

   iii. *Add a new paper to the offerings of a given department, specifying the paper details listed above.*

   iv. *For a given paper (identified by its paper code, e.g. `INFO303`), update the paper details.*

   v. *Delete a paper that is no longer offered.*

   You are not required to discuss issues related to authentication or error cases.

   (a) Write down the resource URL patterns that you would use for your REST web service. Label these (e.g. P1, P2, ...) so you can refer to them later in this question. (4 marks)

   (b) For *each* of the operations i – v above:

   - State the HTTP method that will be used for the operation in your REST API.
   - Identify the URL pattern that the HTTP request will be addressed to (you can simply refer to a pattern you gave in part (a), e.g. "P1").
   - Describe the bodies of *both* the request and response messages (if empty, just state that).
   - Describe any information that will be included in the response message as an HTTP message header rather than in the response body. You do not need to mention the HTTP response code.

   (12 marks)

2. **[The suggested time allocation for answering this question is 35 minutes.]**

   (a) Write a set of Apache Camel route definitions that implement the requirements described in the scenario on page 5.

   You will not be penalised for minor syntactic errors in your route definitions, but you should clearly show each message route, the required processing steps within each route, and the endpoints used. (15 marks)

   (b) Briefly describe any Java domain classes that you would need for this Camel application by explaining *what information* the class(es) would represent, *how* this information would be structured, and *what methods* you would need to implement in the class(es). (4 marks)

3. **[The suggested time allocation for answering this question is 7 minutes.]**

   Describe *two* benefits of developing and deploying an application using the *microservices architecture* compared to a more traditional approach. (4 marks)

4. **[The suggested time allocation for answering this question is 16 minutes.]**

   Explain *three* software engineering benefits of having an OpenAPI specification for a web service. For each benefit, you should include a clear description of *how* the specification is used to achieve that benefit. (9 marks)

5. **[The suggested time allocation for answering this question is 7 minutes.]**

   To be considered as truly "RESTful", a web service should conform to four principles proposed by Roy Fielding: *addressability*, *uniform interface*, *statelessness*, and *connectedness* (the last of these is also referred to using the acronym HATEOAS: "hypermedia as the engine of application state").

   Choose *two* of these principles, and explain what *each* of the two means in practice when designing a web service. (4 marks)

6. **[The suggested time allocation for answering this question is 11 minutes.]**

   (a) Describe the key elements of the GraphQL style of API. Your answer should describe (i) the information that must be specified when designing an API in this style, and (ii) the information that is contained within request and response messages. (3 marks)

   (b) Describe two advantages that GraphQL APIs have compared to REST APIs. Justify your answers, i.e. explain *why* these are advantages. (3 marks)

7. **[The suggested time allocation for answering this question is 14 minutes.]**
   A data warehouse contains two main types of data: *facts* (also known as *measures*) and *dimensions*.

   (a) Explain the purpose of each of these types of data and the relationship between them. Use an example to illustrate your answer. (5 marks)

   (b) Explain what it means for a dimension to have an internal hierarchy, and how such hierarchies can be used to change the way data are viewed. Use an example to illustrate your answer. (3 marks)

8. **[The suggested time allocation for answering this question is 16 minutes.]**

   (a) There are two primary approaches to receiving responses to long-running service operations: *polling* and *callbacks*. Briefly describe the differences between the two approaches. Your answer should include the comparative advantages and disadvantages of each approach. (6 marks)

   (b) What standard web technologies could be used to implement a callback mechanism for receiving responses from long running service operations? (3 marks)

9. **[The suggested time allocation for answering this question is 7 minutes.]**

   Provisioning systems such as Puppet and Vagrant often use the idea of Infrastructure as Code (IaC) to help developers and administrators configure their infrastructure. What specific problems does using IaC help with? (4 marks)

10. **[The suggested time allocation for answering this question is 16 minutes.]**
    Compare and contrast NoSQL DBMSs with "traditional" SQL DBMSs with respect to database structure, data integrity and the kind(s) of data management environments that each is best suited to. (9 marks)

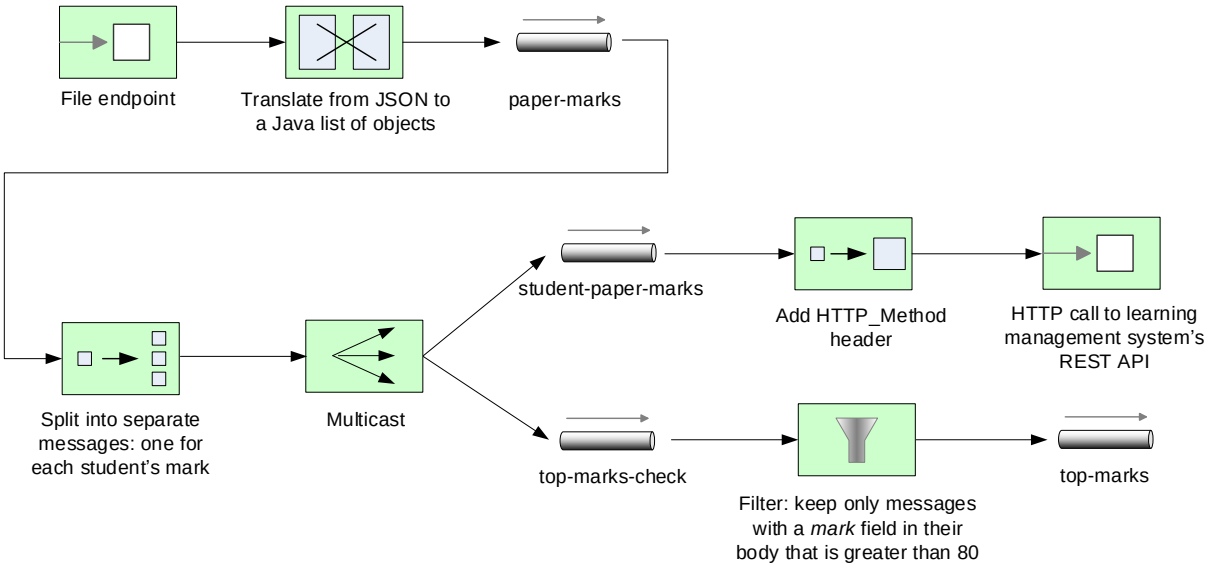
11. **[The suggested time allocation for answering this question is 14 minutes.]**
    Compare and contrast the use of *replication* and *partitioning* (also known as *sharding*) in the context of large-scale distributed databases. Discuss each technique's main goals, how it achieves these, and any disadvantages of the technique. (8 marks)

12. **[The suggested time allocation for answering this question is 7 minutes.]**

    Explain the role of an enterprise architect (EA) in terms of what an EA does and how this brings value to an organisation. (4 marks)

The scenario for Question 2 is uploading a set of final marks for a university paper to a learning management system such as Blackboard. The marks are initially stored in a file containing a sequence of JSON records. Each record contains the fields *studentID*, *paperCode* and *mark*. Your route must perform the following steps, and is illustrated in the diagram below using enterprise integration patterns.

- Read the data from the file using a "file" endpoint with the URI `file:FinalMarks`. This will convert any file in the FinalMarks folder to a Camel message that contains the file contents (JSON data in this case) in its body.
- Convert the message body from JSON into a Java list, and place the resulting message into a queue named *paper-marks*.
- Split each message in the *paper-marks* queue into separate messages, each containing the mark for a single student and paper combination.
- Send each message resulting from the split operation to two queues: one named *student-paper-marks* and another named *top-marks-check*.
- For each message on the *student-paper-marks* queue:
  - Add a message header, with the header's *name* set to the value of the Java constant `Exchange.HTTP_METHOD` and the header's *value* set to "POST".
  - Send the message to an HTTP endpoint with the URL `http://lms.otago.ac.nz/@api/final-marks/`.
- For each message in the *top-marks-check* queue, apply a filter that only passes on messages in which the *mark* field in the message body has a value greater than 80. These messages are put into a queue named *top-marks*. (You do not need to process the messages in the *top-marks* queue: a separate application will aggregate these to inform the appropriate staff member of potential candidates for awards.)



Problem scenario for Question 2

END