ECM3408 Enterprise Computing
Continuous Assessment

Handed Out	Handed In
Mon 1st February 2021 (T2:04)	**Wed 3rd March 2021** (T2:08)

This Continuous Assessment is worth 30% of the module mark. It is an **individual assessment**, to be submitted using BART. Your attention is drawn to the College and University guidelines on collaboration and plagiarism.

1 Scenario

EasyRide is a ride-sharing enterprise based in the UK. The enterprise allows drivers to join a roster by name, giving the rate in pence/km at which they are prepared to offer rides, to change the rate at which they are prepared to offer rides, and to leave the roster. A driver must be authenticated in order to join the roster, to change the rate at which they are prepared to offer rides, or to leave the roster. The enterprise allows riders to specify the start- and end-points of journeys, and to obtain the best driver for the journey. The best driver is calculated by surge pricing, based on the route taken, which is obtained from a mapping service, the driver rates, which are obtained from the roster, and the time of day, which is obtained from the system clock. The surge pricing algorithm doubles the cost if the majority of the journey is to be made on 'A' roads (that is, roads whose designation begins with the character 'A'), doubles the cost again if there are fewer than five drivers on the roster when the journey is to begin, and doubles the cost again if the journey is to begin between 23:00 and 06:00.

2 Assessment

The assessment is organised into three parts.

2.1 Part 1: Microservice Identification (20 % of marks)

Briefly describe a number of microservices suitable for realising the EasyRide ride-sharing service. The description should take the form of a graph whose nodes (circles) are microservices, and whose arcs (lines) are communication channels between them. The nodes should be accompanied by a short description of what the microservice does, and the arcs should be accompanied by a short description of the data flow.

Marking Scheme: Your microservices will be judged on their appropriateness **(8 % of marks)**, coherence **(6 % of marks)** and lack of coupling **(6 % of marks)**. Here, appropriateness means relevance to some part of the EasyRide ride-sharing service; coherence means focus on just one part of the EasyRide ride-sharing service, and lack of coupling means independence from other parts of the EasyRide ride-sharing service.

2.2 Part 2: Microservice Implementation (50 % of marks)

For each microservice that you have identified for realising the EasyRide ride-sharing service, show some `Go` code that implements it. The implementation should provide a Representational State Transfer (REST) interface over the Hypertext Transfer Protocol (HTTP). You may use the Google mapping service to obtain distance data.

Marking Scheme: Your microservice implementations will be judged on the clarity of their source code **(20 % of marks)**, the correctness of their operation **(20 % of marks)**, and the thoroughness of their error handling **(10 % of marks)**. Here, the clarity of source code refers to the layout of functions, statements and expressions, the choice of identifiers, and the use of comments; the correctness of operation refers to the processing of inputs to give outputs; and the thoroughness of error handling refers to the signalling of invalid input, output or processing with appropriate HTTP response codes. You should keep in mind the Single Responsibility Principle, the Law of Demeter, and the Don't Repeat Yourself edict.

2.3 Part 3: Microservice Deployment (30 % of marks)

For each microservice that you have identified for realising the EasyRide ride-sharing service, show a `Dockerfile` that could be used to provision a networked `docker` container providing the microservice, and a command that would start the container. Deploying these containerised microservices together should realise the ride-sharing service.

Marking Scheme: Your microservice deployment will be judged on how well the microservices work together in their networked containers **(30 % of marks)**.

3 Submission

You are expected to submit the following items:

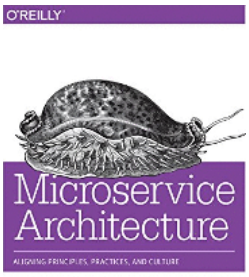
- The source code of your system of microservices;
- PDF version of a listing of the same source code;
- A short report.

All items should be submitted using BART (`https://bart.exeter.ac.uk`) by **midday on Wednesday 3rd March 2021**

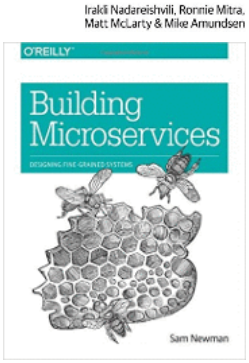
Your code should be clearly commented, and your report should have a maximum of 3 pages (3 sides of A4) which should include

- A graph along with short description of the microservices as mentioned in *Part 1: Microservice Identification*;
- Clear demonstration on how each RESTful operation is implemented, and how any errors that occur are dealt with;
- Sample commands and their outputs to show that you have tested your service.

References



I. Nadareishvili, R. Mitra, M. McLarty and M. Amundsen, *Microservice Architecture: Aligning Principles, Practices and Culture*, O'Reilly, 2016, ISBN 978-1491956250



S. Newman, *Building Microservices: Designing Fine-Grained Systems*, O'Reilly, 2015, ISBN 978-1491950357.