CS3219 S1-AY2020-21: Self-Paced Tasks

This document describes 2 ways you can accumulate a **maximum of 25 marks** for self-paced tasks. which includes an additional 7 marks for early submission or going beyond what is required.

OPTION 1

18 Marks: Accumulated by doing tasks listed in A-F.

7 Marks: Accumulated through options AA-AD listed in the 'Additional Marks' section.

- **Important**: These 7 marks cannot be earned without completing a minimum of 18 marks worth of tasks from A-F.

OPTION 2

25 Marks: Accumulated by doing tasks listed in A-F.

Read the instructions carefully. Send your queries to cs3219se@gmail.com.

Note: Once you complete a task, notify us through a submission at LumiNUS. Each file submission in LumiNUS must contain your **Matriculation Number in the file name**. There is a separate folder for each type of task mentioned below.

Own Time Own Target

Task	Marks	Description	Instructions
A	**3**	**TASK:** Deploy a simple web server using Nginx running in a Docker container. **LEARNING OBJECTIVE:** To understand how containerization works and what its advantages are. **MARKING SCHEME:** ● Demonstrate ability to write simple Dockerfile (1 mark) ● Setup Nginx and run a reverse proxy (1 mark) ● Demonstrate ability to use Docker Hub registry to keep track of images (1 mark) **RESOURCE LINKS:** https://github.com/docker/labs/blob/master/beginner/chapters/setup.md https://docs.docker.com/get-started/ https://docs.docker.com/docker-hub/ https://www.nginx.com/resources/wiki/start/ https://www.nginx.com/resources/library/complete-nginx-cookbook/	Submit a PDF document as proof of completion. The document should contain the following information: ● Student Name and Matriculation Number. ● Link to the GitHub repository. ● Instructions on how to run the Docker container. ● Screenshot(s) of Docker Hub registry ● Any other relevant learnings.

<table>
<tr><td>B1</td><td>3</td><td><u>TASK:</u> Implement a simple Javascript backend and REST API to GET, POST, PUT, DELETE (example: an API for a quotes).

<u>MARKING SCHEME:</u>
● Successful GET, POST, PUT, DELETE API calls on localhost using Postman (1.5 marks)
● Successful GET, POST, PUT, DELETE API calls to deployed endpoints using Postman (1.5 marks)

<u>RESOURCE LINKS:</u>
https://expressjs.com/en/starter/hello-world.html
https://medium.com/@dinyangetoh/how-to-build-simple-restful-api-with-nodejs-expressjs-and-mongodb-99348012925d</td><td rowspan="2">Submit a PDF document as proof of completion. The document should contain the following information:
● Student Name and Matriculation Number.
● Link to the GitHub repository.
● Instructions on how to:
○ Run the API locally (B1)
○ Access deployed API (B1/B3)
○ Run tests locally and via Travis (B2)
○ Set up frontend (B4)

NB: For these four tasks, you may choose to attempt any number of them (i.e. 1-4) but do note that each subsequent task builds on the previous one. Include the</td></tr>
<tr><td>B2</td><td>3</td><td><u>TASK:</u> Write simple test cases for API and use a CI tool (Travis, etc) to automate testing.

<u>LEARNING OBJECTIVE:</u> To understand the importance of testing and CI tools in simplifying the process of testing.

<u>MARKING SCHEME:</u>
● Demonstrate use of testing for API using Mocha/Chai, or any other testing framework (1.5 marks)
● Demonstrate ability to use Travis or any other CI tool to automate testing (1.5 mark)

<u>RESOURCE LINKS:</u>
https://mochajs.org/#getting-started
https://medium.com/@asciidev/testing-a-node-express-application-with-mocha-chai-9592d41c0083 https://dev.to/bushraalam/introduction-to-testing-with-mocha-and-chai-57po</td></tr>
</table>

B3	**3**	**TASK:** Use CD tool for automated deployment to a serverless service. **LEARNING OBJECTIVE:** To understand the power of automated deployment, and serverless tools in the creation of APIs or microservices. **MARKING SCHEME:** ● Demonstrate ability to automatically deploy tool to a serverless service (e.g. AWS Lambda or Google Cloud Functions) (3 marks) **RESOURCE LINKS:** https://docs.travis-ci.com/user/deployment/lambda/ https://www.cloudflare.com/learning/serverless/glossary/serverless-microservice/ https://microservices.io/patterns/deployment/serverless-deployment.html https://martinfowler.com/articles/serverless.html	*aforementioned instructions on how to run each task based on those that you have attempted.*
B4	**3**	**TASK:** Build a frontend SPA using Vue or any other framework. **LEARNING OBJECTIVE:** To understand Model-View-Controller in the context of real-life frameworks. **MARKING SCHEME:** ● Demonstrate ability to interact with the API using the frontend (2 marks) ● Implementation of style e.g. using Bootstrap (1 mark) **RESOURCE LINKS:** https://vuejs.org/v2/guide/ https://vuejs.org/v2/cookbook/using-axios-to-consume-apis.html	

C	**3**	**TASK:** Propose a possible microservices architecture for ChairVisE (or any other project/existing software) **LEARNING OBJECTIVE:** To understand how a monolithic application can be broken down into multiple microservices. **MARKING SCHEME:** ● Broad overview of architecture in diagram form: which feature sets are grouped together (1 marks) ● Rationale for proposed architecture (1 marks) ● High-level implementation detail e.g. communication interfaces, framework for each microservice (1 mark) **RESOURCE LINKS:** https://martinfowler.com/articles/microservices.html *NB: If your project team is revamping ChairVisE3.0 with a microservices architecture implementation, you should attempt this task with some other software (new or existing). You will receive no credit if you attempt it with ChairVisE3.0.*	Submit a PDF document as proof of completion. The document should contain the following information: ● Student Name and Matriculation Number. ● Microservices architecture overview diagram. ● Rationale for microservices architecture. ● High-level implementation details.
D	**6**	**TASK:** Create a 3 node Apache Kafka cluster using Docker that demonstrates the Pub-Sub messaging, with a Zookeeper ensemble created to manage the Kafka cluster. **LEARNING OBJECTIVE:** To have hands-on experience implementing a messaging service and managing failover in containers. **MARKING SCHEME:** ● Successful implementation of Pub-Sub messaging system using Apache Kafka (3 marks)	Submit a PDF document as proof of completion. The document should contain the following information: ● Student Name and Matriculation Number. ● Link to Github repository.

		• Successful management of the failure of the master node in the cluster, i.e. another node takes over as master node (3 marks) **RESOURCE LINKS:** • https://kafka.apache.org/intro • https://hackernoon.com/thorough-introduction-to-apache-kafka-6fbf2989bbc1 • https://medium.com/rahasak/kafka-and-zookeeper-with-docker-65cff2c2c34f	• Instructions on how to set up and run the Kafka cluster.
E	**6 (3 marks per post)**	**TASK:** Write and publish two technical blog posts on any of the software engineering principles and patterns that you have learnt in class. **LEARNING OBJECTIVE:** To reinforce concepts learnt in the module through teaching. **MARKING SCHEME:** Per blog post: • Clear, precise language used to explain the principle or pattern (1 mark) • Effective use of diagrams (1 mark) • Providing a concrete example (hypothetical or real-world) where such a pattern or principle is utilized (1 mark) While you should not be too verbose, your post should have a minimum length of 400 words. For the publishing platform, you can consider using Medium.	Submit a PDF document as proof of completion. The document should contain the following information: • Student Name and Matriculation Number. • Link to each blog post.

F	**3**	**<u>TASK:</u>** Suggest a change or addition to the lecture or tutorial material. For this task you may do any of the following: 1. Propose an existing topic you would like covered with greater depth 2. Propose a new topic to be covered in the module 3. Propose the removal of an existing topic **<u>LEARNING OBJECTIVE:</u>** To encourage students to critically examine the existing course curriculum and explore content outside of what is taught in the classroom. **<u>MARKING SCHEME:</u>** ● Well-elaborated rationale for the proposal given (1 mark; 3 marks if choosing option 3 "removal of an existing topic") ● Brief explanation of the pedagogy, i.e. what you would teach about this new or more in-depth topic and how you would teach it (2 marks; NA for option 3 "removal of an existing topic")	Submit a PDF document as proof of completion. The document should contain the following information: ● Student Name and Matriculation Number. ● Write-up on the change or addition to the material that covers the points in the marking scheme.

Additional Marks

Early submission		
AA	**Max 3 marks**	Either: ● 2 marks for completing 9 marks worth of tasks A-F by the end of week 7 (Sunday, 4th October 2020, 2359H). ● 1 mark for completing additional 9 marks worth of tasks A-F by the end of week 10 (Sunday, 25th October 2020, 2359H). Or: ● 2 marks for completing 18 marks worth of tasks A-F by the end of week 10 (Sunday, 25th October 2020, 2359H).
Extra effort		
AB	**Max 3 marks**	● Lecturer and tutors will be on lookout for notable submissions that are worthy of mention. This will be announced during lectures or through LumiNUS announcements. ● You score an additional 1 mark every time your submission gets featured by the tutor or lecturer, up to a maximum of 3 times (for a maximum of 3 marks).
AC	**Max 6 marks**	● Up to 6 marks for contributions or pursuits outside of class (must be dated between 10 August 2020 and 13 November 2020). These can include: o Merged Open Source PRs o Certifications related to Software Engineering o Application of module learning in other areas (which can be either in other modules or personal projects) ● For each of the above demonstrated, one mark will be awarded up to a maximum of 6 marks (e.g. 3 merged open source PRs and 2 software engineering certifications will get you 5 marks). ● If unsure about what counts towards this, clarify with the lecturer or tutors as soon as possible. ● Submit a PDF document with your Name and Matriculation Number that details your contributions or out-of-class pursuits. This includes any supporting documentation for verification (e.g. a link to the software engineering certification you have achieved, or the merged open source PR that you worked on).

Bonus project task		

AD	**1 or 2 marks**	• Deploy module project (DIY web application or ChairVisE) using Google App Engine (or any other platform) for demonstration and to fully complete an entire SDLC.
		• Please add the link to the deployed application in your project report. Only a fully usable, deployed web application will earn you the marks.
		  ○ Also include the names of those who worked on the deployment. This will determine the number of marks that will be given.
		  ○ If 2-4 team members have worked on the deployment, they will receive 1 mark each. If only a single member worked on the deployment, only that member will receive 2 marks.
		• Use the following resource links if needed:
		https://www.baeldung.com/spring-boot-google-app-engine
		https://codelabs.developers.google.com/codelabs/cloud-app-engine-springboot/
		• *NB: The use of Google App Engine or other cloud platforms may often require payment. However, they do provide free credits or free tier versions of some of their services for limited periods of time. Please review these and ensure that you can utilize them to avoid any unnecessary costs.*

Send your queries to cs3219se@gmail.com.