# Executive Summary

|  |  |
| --- | --- |
| **Assignment Title** | **Building an Enterprise Solution Using Microservices Architecture** |
| **Team Size** | 5-6 persons |
| **Grade Percentage** | 30% of all assessments |
| **Due Dates** | |
| **Proposal** | **Before the Week 9 class**  Your team must have prepared a proposal to describe your proposed scenario (e.g. Taxi Application, Food Delivery, etc.) and draft solution design. The details of the proposal are under the [Deliverables-Proposal](#_30j0zll) section.  **During the Week 9 class**  Each team gets **20 minutes** to present and discuss your scenario and draft solution design with the instructor(s) face-to-face in the classroom. |
| **Presentation** | **30 minutes before the start of your respective class in Week 13,**  submit your **presentation slides** on eLearn. See [Deliverables-Presentation](#_1fob9te) section for details.  E.g., 0815 class 🡪 0745hrs  1530 class 🡪 1500hrs  **Present & demonstrate** your scenario and solution in class (**within 12 minutes**) followed by Q&A |
| **Documentation and Executables** | **By the end of your respective class in Week 14,**  submit **a final report and your code** on eLearn. See [Deliverables-Documentation&Executable](#_3znysh7) section for details.  E.g.,   0815 class 🡪 1130hrs  1530 class 🡪 1845hrs |
| **Submissions** | All required submissions are to be done through eLearn. We will take and mark the latest submission on eLearn but apply **heavy penalties for late submissions**. See [Deliverables-Submission](#_2et92p0) section for details. |
| **Intra-Team Peer Evaluation** | **On a CASE-BY-CASE basis**   * If there is any issue(s) within the team, you **MUST** let us know the problem(s) **AS AND WHEN** the problem(s) surfaces. If you have issues that persisted throughout the project and inform us about it only near the end of the project, we are afraid that there will be **not much we can do to resolve the issues at that late stage**. * There is no official intra-team peer-evaluation for every team. We will investigate on a **CASE-BY-CASE** basis and make appropriate decisions on penalties to the member(s) who did not do their part for the project. |

# Learning Outcomes

With the emergence of new technologies and evolution of existing ones, organizations are changing the way they build enterprise solutions. Rather than build monolithic applications, the current emphasis is on building solutions by leveraging the microservices architecture. This approach to building solutions using microservices follows the Service Oriented Architecture (SOA) paradigm, where applications are structured as assemblies of loosely coupled (micro)services.

⬥ ⬥ ⬥

In this assignment, your team must follow **a methodical approach** to come up with **an enterprise solution** based on the **microservices architecture** for **a business scenario** (e.g., Taxi Booking, Food Delivery, etc.) using **appropriate technologies and tools** (e.g. Python, REST, RabbitMQ, WampServer, Docker, HTML, Javascript, Visual Studio Code, etc.).

# Minimum Requirements

* Your business scenario and solution must implement and walkthrough **at least 3 user scenarios**
  + Sample user scenarios and data entities involved for Taxi booking

(1) Passenger sends a booking request…

* + - Passenger, Booking, …

(2) Driver is notified and accepts a booking…

* + - Driver, Booking, Notification, …

(3) Passenger pays for a ride…

* + - Passenger, Payment, Booking, …
* Your scenario and solution must have **a minimum of 3 microservices for 3 different data entities**. You may use any programming language to implement the microservices.
* Each microservice can have more than one operation (CRUD).
* Each microservice must have exclusive access to its own data store (if it has a data store). A data store can be a file or a DB table or cloud storage.
* At least one of the microservices has to **use a DB** as its data store.
* At least one of the microservices should be **reused** across different user scenarios.
* The communication between (micro)services must **use both HTTP and AMQP** technologies. You can choose to implement any appropriate communication pattern(s).
* Your application must have **a web based Graphical User Interface** which can be developed in any programming language (e.g. Javascript, PHP, JSP/Servlet, Python with Django, etc)
* **Use JSON data** in some of your microservices and/or Web UIs.
* **Use Docker** in your solution in a way suitable for your scenario.
* Implement at least one thing that is **beyond the labs** (something not covered in the labs or something considered as advanced topics in the labs that you need to research and implement on your own effort)
  + See below for [examples beyond the labs](#_tyjcwt)

# Deliverables

## Proposal

You must prepare a proposal before the discussion session with your instructor(s). Discussing with the instructor(s) and your classmates early is an opportunity for you to get feedback on your scenario choice and solution design and allows ample time for implementation. Your scenario and solution **can still be changed or refined after the discussion**.

A template of the proposal is included with this requirement document for your reference. You MUST prepare your proposal following the "**Proposal Template**" file for the discussion.

During the 20-minute discussion session in class, your team **must**

* Present (informally) the proposal documents to your instructor(s);
* Explain your chosen scenario, simplification and/or assumptions, solution overview diagrams, design choices, possible uses of different technologies, and potential challenges in implementing the scenario and design;
* Answer any question about your scenario and solution design and/or take notes of it for improvement in your later design refinement and implementation stage.

All the team members should be present for the discussion session. You are not required to submit the proposal. **The proposal is not graded, but very poor preparation of the proposal documents will get penalty towards your final assignment marks**.

## Presentation (20 marks)

* **Presentation slides**
  + You must submit your Presentation slides (in MS PowerPoint format) **before the presentation submission deadline**. You may draw diagrams following the format of the proposal/report templates.
  + You must have **at least 1 slide on what your team has done beyond the labs**.
* **In-class presentation and demonstration (12 minutes)**
  + You must use the submitted presentation slides to present your scenario and solution;
  + You must reserve ample time within the time limit for the demonstration of your solution;
  + You will not be allowed to present beyond the time limit regardless of whether you have finished your demonstration, which may lead to a failure mark for your demo portion.
* **A Q&A session** will follow each presentation and demonstration. Every member of your team should be prepared to answer questions from all persons watching your presentation and demonstration, including your classmates and your instructor(s).

## Documentation & Executables (10 Marks)

* **Report**
  + You must submit a report (in MS Word format) **before the report submission deadline**;
  + The page limit is **max 6 pages**, excluding the cover page(s). Any page in the report that is beyond the page limits will not be marked.
  + You **must** write the report following the “**Report Template**” given to you.
  + Your report must be presented in a coherent way and the flow must be consistent.
* **Executables**
  + You must submit all of your code (e.g., .py, .html, .js files), batch scripts (e.g., .bat, .sh, and Dockerfile(s) used in your solution), configurations (e.g., particular database accounts and passwords used), and data files (e.g., pictures, .json, .csv, exported database .sql files) **in one .zip file, separated from the report document**. Do **NOT include files that can be generated** from your code/scripts/configurations/data (e.g., the Docker images). If your zip exceeds 100MB, consult your instructor(s) before submission.
  + You must also write a readme file, **name it README.txt**, giving instructions on how to set up and run your solution using your submitted .zip file (e.g., say so if your solution needs to run WampServer, or use a special account, or install additional python modules or other software, etc.).
  + The .zip file is not explicitly graded, but it will be used to verify against your solution as described in the report if required. Penalty applies if no submission.

# Submissions

* All submissions must be done electronically through the right eLearn Assignments drop-boxes; any other medium for submission (e.g., email, printout) will not be accepted unless otherwise specified by your instructor(s).
* All submission deadlines must be strictly adhered to. You are strongly encouraged to submit early, taking potential network congestion into account.
* **Heavy penalties for late submissions** are as follows:

|  |  |
| --- | --- |
| within 1 hour (one second late is late) | 10% deductions off the marks you deserved |
| each subsequent hour | Penalty will double (i.e. 20%, 40%, 80%, and finally 100%) |

# General Guidelines

* Your business scenario can be imaginative or futuristic, but it must be legal and morally suitable for use in academic settings. E.g., scenarios suggesting illegal gambling will not be allowed.
* State your simplification and/or assumptions, if any, for your scenario and solution.
* Use simple code for the user interface- Functionality is more important than the look-and-feel. For example, a fancy UI is nice, but if it does not help you in any way other than just looking pretty, it is not going to earn you a lot of marks or any at all.
* You may reuse any code in the labs with and without modifications.
* You may use any technology and tool to implement your solution, not limited to the technologies and tools covered in the course. Whether those technologies and tools are considered as going beyond the labs depends on how you use them and is at the discretion of the instructor(s).
* To cater for some unfortunate situations (e.g., your demo works before the class but it fails during actual demo in class), you may prepare a video recording of your demonstration, or a sequence of screen captures before the class as a backup.

## Some examples beyond the labs

* Invoke external REST APIs (e.g. Facebook, Twitter, Google Maps, PayPal, etc)
* Implement functionality to handle business logic related errors or exceptions
  + E.g., what happens if inventory is not enough for a new order; what happens if no driver responses to a taxi booking request within a time limit;
  + Handling of errors/exceptions from code (e.g, try...except in python, or try...catch in JS) is not counted under “beyond the labs” for this assignment.
* Implement some complex (micro)services that orchestrate simple (micro)services.
* Use API gateway in the solution in a reasonable way.
* Deploy Docker containers containing your microservices to a cloud platform.
* Enable communication among microservices running on different physical machines.   
  If you are using WLAN-SMU, peer to peer network communications works in SIS building ONLY.   
  If you are using your own hotspot, there are no restrictions.

## Penalty Situations

* Fail to show your scenario or draft solution design during the discussion session.
* Late or no submissions on eLearn.
* Presentation of a version of your slides different from the latest version you submitted on eLearn before your presentation.
* Change your slides or code while another team is presenting. We will give all teams about 15 minutes at the start of the class to set up (e.g., starting RabbitMQ, WampServer, various Flask services, and needed cmd windows or web UIs, resetting values in database, testing with the projector, etc.) After which, **no one is allowed to be “busy” working on his/her own work while others are presenting**.
* Run over time during the presentation due to your own problem (e.g., lack of preparation or rehearsal, lack of testing of your demo code, etc.).
* Fail to demonstrate your solution during the presentation, which will lead to a failure mark for your demonstration portion right away.
* The report and its appendix exceeds the page limits.
* Poor inter-team peer evaluation (such as low points and feedbacks from teams evaluating you).
* Poor intra-team peer evaluation (if any).
* Any other inappropriate situation at the discretion of your instructor(s).

## ***About using external services***

* DO NOT misuse the services kindly provided by the service providers listed below.
* As a rule of thumb, use your official SMU email to create only **ONE** test/demo account, and avoid unnecessary requests to those servers.
* Test only a few times with the actual external server and setup (as per what you have registered), and after that, reuse and simulate the replies or outputs from the servers during your further development and testing so as to reduce the hits on those live servers. Only switch back to use the actual servers when you are close to the demonstration.

E.g., I have registered for an email service via a web service. After testing it via code to send emails via this web service once or twice and knowing its behaviour, I replace the actual code with some simulation code to mimic the sending and receiving of emails to get things still working (although not like production). Only when it is nearer to the presentation, I change the code back to the actual code and test it a few more times with the live servers before the demo.

* It can happen that the external services become unavailable during the demo. To cater for such unfortunate situations, you may prepare a video recording of your demonstration with those live services before the presentation.

### Sample External Service Providers

|  |  |
| --- | --- |
| Google | <https://developers.google.com/> |
| Amazon | <http://aws.amazon.com/> |
| Xignite | <https://www.xignite.com/financial-data-apis> |
| Facebook | <https://developers.facebook.com/docs/> |
| PayPal | <https://developer.paypal.com/docs/checkout/> |
| YouTube | <https://developers.google.com/youtube/> |
| Twitter | <https://developer.twitter.com/en/docs> |

# Marking scheme

Here are some (but not restricted to) of the criteria that may determine your grade for the assignment:

## (1) In Class-Presentation with Demo

* Is the scenario clearly explained?
* Is there a smooth flow of the ideas throughout the presentation?
* Are the slides well designed and clearly readable?
* Is the presentation well-paced?
* Are there sufficient details provided in the presentation to help the audience understand the diagrams?
* Does the demo clearly demonstrate the scenario discussed?
* Does the demo work without any hiccups?
* Are the user interactions and inputs/outputs clearly visible?
* Was the team able to answer questions asked by the audience?
* Did the presentation show the team’s understanding of various concepts relevant for the course?

### Technical Depth

* Have the minimum technical requirements as per the assignment description been fulfilled?
* Are there any technical aspects that went beyond the labs?
* How relevant are the technical implementations for the scenario?
* How complex are the technical implementations (including those beyond the labs)?

### Rubrics

|  |  |
| --- | --- |
| **Marks** | **Details** |
| <11 | Presentation with Demo sloppy. Bare minimum technical depth. |
| 11-13 | Presentation with Demo is of reasonable standard. Reasonable technical depth |
| >13-16 | Presentation with Demo is good. Good technical depth. |
| >16-18 | Presentation with Demo is very good. Very good technical depth. |
| >18 | Presentation with Demo is exceptional. Exceptional technical depth. |

## (2) Report

* Is the report well-structured (including professionalism)?
* Does the report explain the scenario clearly?
* Does the report clearly describe the technical implementation?
* Are there discrepancies between the different sections of the report?
* Are the diagrams clear and consistent with the scenario and technical implementation?
* Are the technical aspects that go beyond the labs clearly explained in the report?
* Are the terminologies used in the report correct?

### Rubrics

|  |  |
| --- | --- |
| **Marks** | **Details** |
| <5 | Report is sloppy. |
| 5-6 | Report is of reasonable standard. |
| >6-7 | Report is good. |
| >7-8 | Report is very good. |
| >8 | Report is exceptional. |

## (3) Penalty Situations at the discretion of the instructor(s).