

Persistence and Analytics Fundamentals Assessment

Date: Friday Mar 03 2023

Assessment Time: 0900 - 1700 (including meal breaks)

Overview

There are **9 tasks** in this assessment. Complete all tasks.

Passing mark is **65% (92 marks)**. Total marks is **142**.

Read this entire document before attempting the assessment. There are 11 pages in this document.

Application Overview

You will be developing a funds transfer application. The application has the following 2 views

- View 0 - the funds transfer page. This is also the 'landing page' of your application
- View 1 - successful transfer

The flow to the application is shown in Figure 1 below

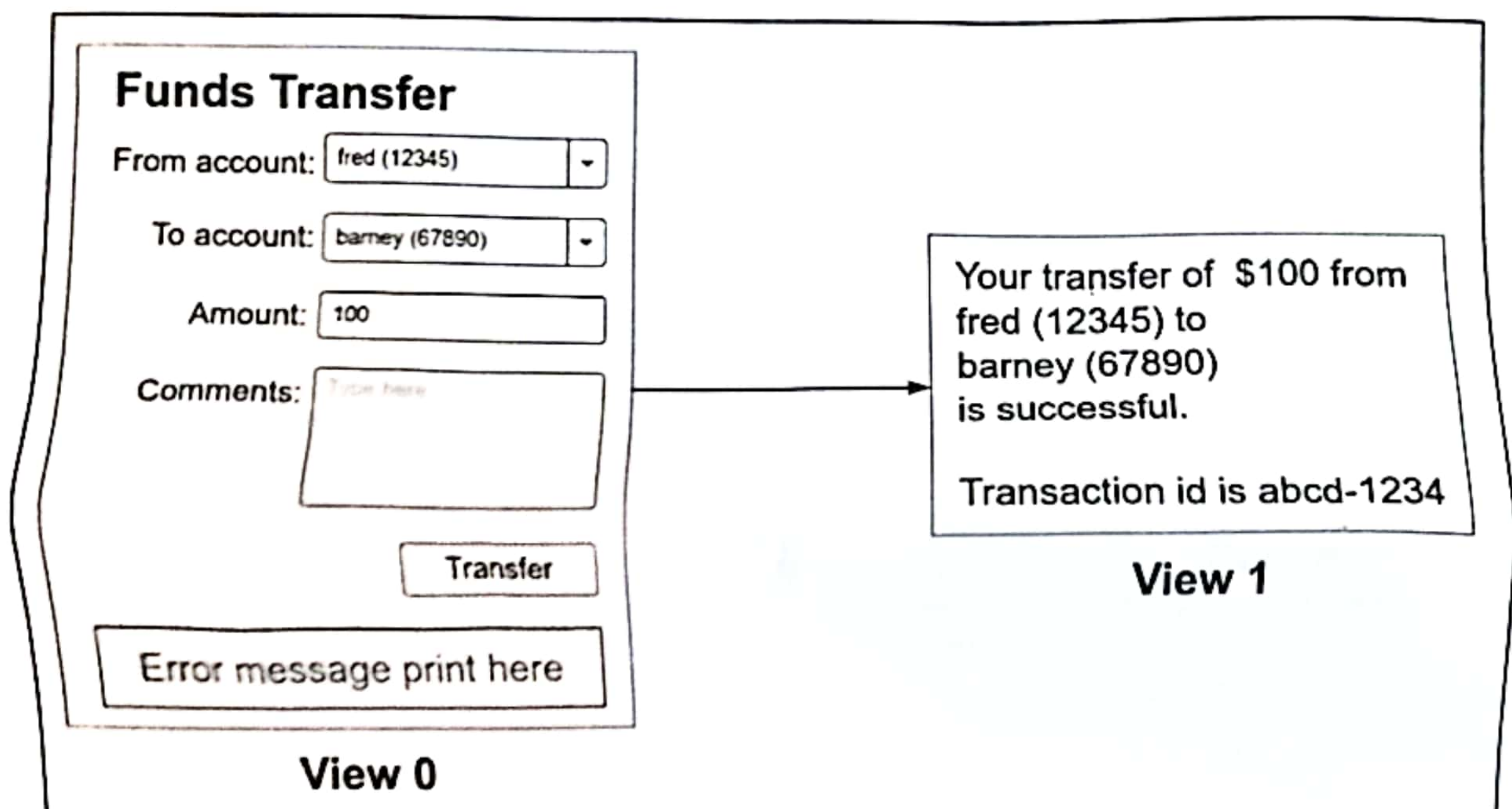


Figure 1

Task 1 (6 marks)

Generate a Spring Boot application; add the following additional dependencies

- Validation
- JSON-P
- MySQL driver
- Jedis driver

You are free to add additional dependencies to the `pom.xml` file.

create a git repository for the assessment. This repository must initially be a **PRIVATE** repository. Click on the 'Private' radio button when you create the repository.

All subsequent work must be saved in this repository. Do not submit more than 1 repository for your final assessment submission.

Once you have set up your repository, you should commit and push your assessment directory. Do not wait until the end of the assessment.

Make your repository **PUBLIC** after 1700 Friday Mar 03 2023 so that the instructors can access your work.

IMPORTANT: your assessment repository is PRIVATE and should only be accessible to yourself and nobody during the duration of the assessment and is only public **AFTER 1700 Friday Mar 03 2023.** If your work is plagiarised by others before the end of the assessment, you will be considered as a willing party in the aiding and abetting of the dishonest act.

Task 2 (6 marks)

Create a project on Railway; in your project provision a MySQL and a Redis database.

Configure your Spring Boot application to use the MySQL and Redis database that you have provisioned.

Sensitive information like the password should not be hard coded into your Spring Boot application.

Task 3 (18 marks)

The file `data.csv` is a comma separated containing bank account details.

Column name	Description
account_id	The bank account number, mandatory
name	Account holder name, mandatory
balance	Account balance with 2 decimal place, mandatory

Table 1

Create a file called `schema.sql` in your Spring Boot project (created in Task 1) and add the following SQL statements

- Create a database called `acme_bank`
- Create a table called `accounts` with the information from Table 1
- Populate the `accounts` table with the account details from the `data.csv` file. Write SQL insert statements in `schema.sql` file.

Use `schema.sql` to create the database in your MySQL database.

You should now have a database with a table called `accounts` populated with data from `data.csv` file.

Hint: for better productivity, you should use local (on your notebook) instances of MySQL and Redis while you are doing your assessment.

Task 4 (38 marks)

View 0 (Figure 2), the landing page, has the following elements (Table 2)

Funds Transfer

From account:

fred (12345)

To account:

barney (67890)

Amount:

100

Comments:

Type here

Transfer

Error message print here

Figure 2 View 0

From account	This is the source account where the funds should be credited. This is a drop down list of the accounts from the <code>accounts</code> table. You should display the account holder's name and the account id eg. fred (V9L3Jd1BBI)
To account	This is the destination account where the funds will be deposited. Also a drop down list of accounts from the <code>accounts</code> table. You should display the account holder's name and the account id eg. fred (V9L3Jd1BBI)
Amount	The amount to transfer from the source to the destination account. You should allow for amount with 2 decimal place eg 100.50
Comments	Allow the user to enter comments regarding the transfer
Error message	The error message area. Any error message

	related to the transfer should be printed here. If there are no errors, then this area should be blank.
Transfer button	Button to perform the transfer

Table 2

Write View 0. Make this your application's 'landing page'.

The From and To account drop down **account details must be dynamically generated from the records in the accounts table**; you cannot hard code it into View 0.

You are free to layout View 1 as long as it contains all the information specified in Table 2.

When the Transfer button is pressed, View 0 makes the following request to Spring Boot with the transfer details

POST /transfer

Content-Type: application/x-www-form-urlencoded

Task 5 (26 marks)

Write a controller called `FundsTransferController` (in a package of your choice) to handle the `POST /transfer` request from Task 4. The funds transfer controller should perform the following checks before performing the transfer

- C0 - Both the accounts from and to should exists
- C1 - The length of the account id should be 10 characters
- C2 - The from account should not be the same as the to account viz. you cannot transfer back to the same account
- C3 - The transfer amount is not 0 or a negative number
- C4 - The minimum transfer amount is \$10
- C5 - The from account should have sufficient funds to perform the transfer viz. the from account balance should be larger than than transfer amount

If any of the above checks fail, the controller should redisplay View 0 with an appropriate message to be displayed in the error message area (Figure 2 Error message print here).

View 0 should also redisplay all the transfer details viz the user should not have to re-enter all the transfer details.

Task 6 (20 marks)

If the transfer passes all the checks in Task 5, proceed with the funds transfer.

Generate a transaction id for the transfer. Use the Java UUID class to generate a transaction id; use the first 8 characters of the generated UUID as the transaction id.

Perform the funds transfer by withdrawing from the 'from' account and depositing the transfer amount into the 'to' account. The update must conform to ACID properties to ensure that the records are not left in an inconsistent state in the event of an error during the updates.

If the transfer is not successful, redisplay View 0, with the appropriate error message from the exception in the designated error message area. View 0 should also redisplay all the transfer details so that the user should not have to re-enter all the transfer details.

The funds transfer should be performed through a Spring Boot service called FundsTransferService; database access should be abstracted into a Spring Boot repository called AccountsRepository. You can create the classes in any package of your choice.

Task 7 (14 marks)

If the transfer is successful, log the transfer details into the Redis database as a JSON document. The JSON document should have the following structure

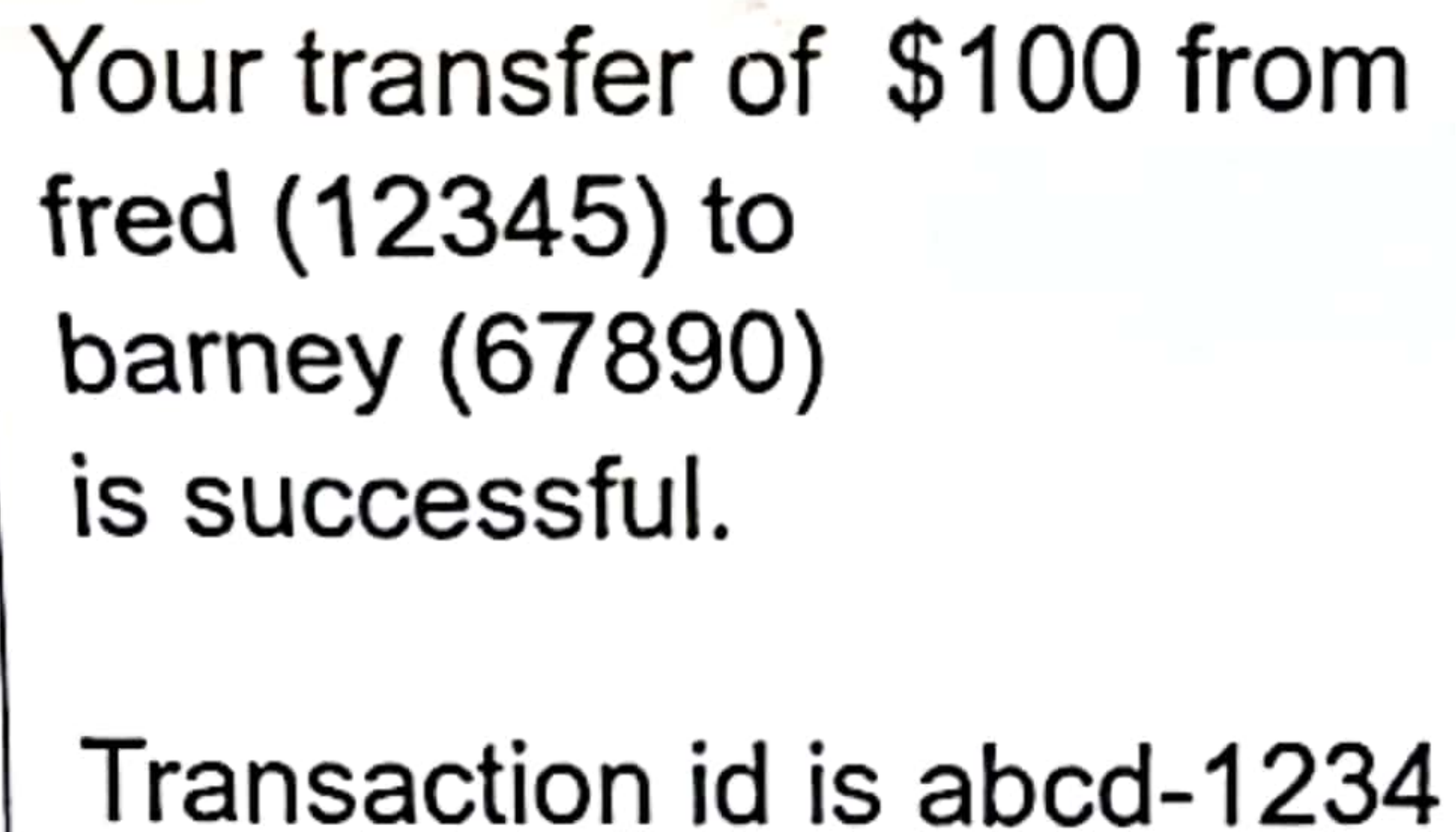

```
{  
  "transactionId": <transaction id>,  
  "date": <transfer date as string>,  
  "from_account": <from account id>,  
  "to_account": <to account id>,  
  "amount": <transfer amount>  
}
```

Use the transaction id as the key for the log entry.

Create a service called `LogAuditService` to perform this logging. You can create the class in any package of your choice.

Task 8 (8 marks)

Once the transaction has been successfully logged to Redis, display View 1 (Figure 3)



Your transfer of \$100 from
fred (12345) to
barney (67890)
is successful.

Transaction id is abcd-1234

Figure 3 View 1

View 1 should contain the following mandatory information:

- From account name
- From account id
- To account name
- To account id
- Amount transferred
- Transaction id (generated in Task 6)

The following flowchart (Figure 4) summarises the funds transfer process

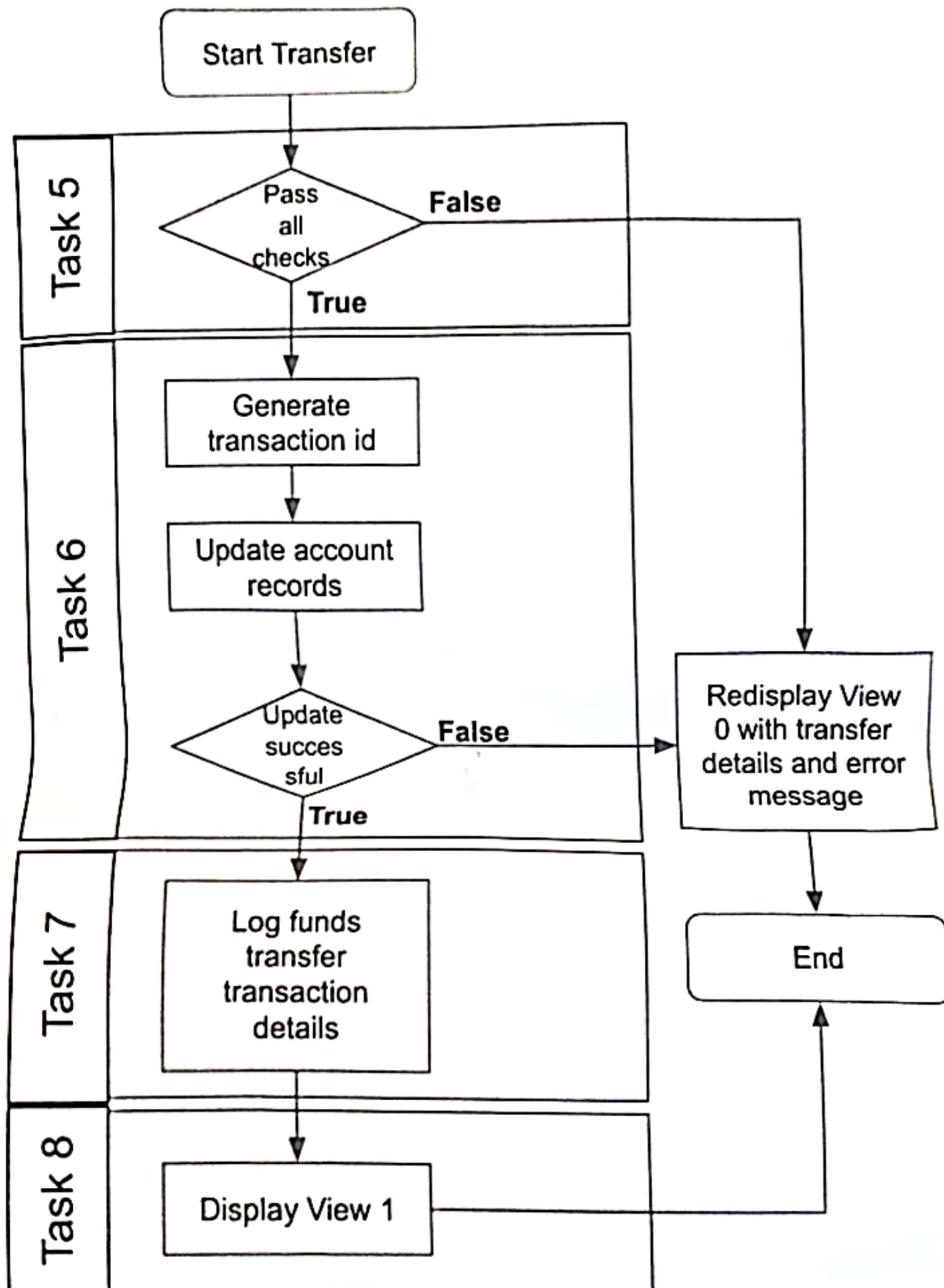


Figure 4

Task 9 (6 marks)

Deploy your application to Railway. The deployment should be based on any commits on or before **1700 Friday Mar 03 2023**.

Do not undeploy your application before 2359 Friday Mar 17 2023.

For JDK19 deployment, add the environment variable NIXPACKS_JDK_VERSION to your Railway service; set the value of NIXPACKS_JDK_VERSION to 19. See <https://nixpacks.com/docs/providers/java>

Submission

You must submit your assessment by pushing it to your repository to either GitHub, GitLab or BitBucket.

Only commits on or before 1700 Friday Mar 03 2023 will be accepted.
Any commits **after 1700 Friday Mar 03 2023** will not be accepted. No other form of submission will be accepted (eg. ZIP file).

Remember to **make your repository public after 1700 Friday Mar 03 2023** so the instructors can review your submission.

After committing your work, post the following information to Slack channel #03-paf-submission

1. Your name (as shown in your NRIC)
2. Your email
3. Git repository URL
4. Railway deployment URL. Please do not undeploy your application from Railway or tear down your MySQL and Redis database until after **2359 Friday Mar 17 2023**

It is your responsibility to ensure that all the above submission requirements are met. Your assessment submission will not be accepted if

1. any of the 4 items mentioned above is missing, and/or
2. your information did not comply with the submission requirements eg. not providing your full name as per your NRIC, incorrect email, forgetting to post the Railway deployment URL, etc, and/or
3. the repository is not public after **1700 Friday Mar 03**

Academic Integrity

This is an open book assessment. You may search the Internet for resources or use reference books during the assessment. The assessment must be your own work. You cannot ask a third party to write any part of this assessment or use AI tools such as ChatGPT to generate output and submit it as part of your assessment. This will result in an automatic disqualification from the assessment.

The NUS ISS takes a strict view of cheating in any form, deceptive fabrication, plagiarism and violation of intellectual property and copyright laws. Any student who is found to have engaged in such misconduct will be subject to disciplinary action by NUS ISS.

You are to ensure the integrity and working condition of your PC/notebooks (eg. wireless/internet connection, battery, screen, accidents like water spillage) during the assessment. NUS ISS will not accept any of these as a reason for deferring or retaking your assessment. accept any of these as a reason for deferring or retaking your assessment.