**Accounting System**

Design a web application for accounting system with below features using technologies mentioned in Section A:

* Login page to the Account system portal with validations on the user credentials
* Password validations: Minimum 8 long, must be alphanumeric with at least 1 special character & 1 alphabet in caps
* User name cannot contain special characters except: underscore ‘\_’, hyphen ‘-‘
* Customer A from Bank A (an account holder in Bank A) can initiate a payment < ZAR 100000.00

This amount must be transferred to Customer B (an account holder in Bank B)

* Make multiple payments from Customer A to Customer C (an account holder in Bank B) in a single day & on multiple days (future dated till 1 month)
* Must have validation on account balance while initiating payment
* Must have validation on the date to not be a weekend (Saturday or Sunday)
* Generate & make payment via batch payment file (From customer A with the batch containing 10 payments to 10 different account holders in Bank B)
* Must have validation on the sender being the same account number
* Validation that the date of payment to all the beneficiaries in the file must be same
* Use Spring batch
* Customer must be able to generate Statements displaying the no. of transactions performed in a single day as well as over a date range (user must have an option to save it in the pdf or text format)
* Create a back-office system which will also be a web application to onboard customers
* Back office system must have login page with login validations (same validations as mentioned in #1.1 & #1.2) followed by pages with below features.
* This application must be a standalone app which can create a new account (account number must be 13 long) while validating account holder details like ID proof number (must be 15 long with no special characters) & Passport number (must be 8 long & alphanumeric with no special characters), validate email ID (must be valid email id format), date of birth (must be dd/mm/yyyy format), age (must be number up to 3 long).
* Must have 2 level authorization where capturer captures the details & authorizer authorizes the account after which it becomes active & available for transaction
* Report generation capability must be built where the capturer can generate a report showing the list of accounts active or inactive (user to have an option to save the report in csv or pdf format)
* Create unit test cases to test the functionality
* Create flow diagram document

**Section A:**

Java, Spring, Spring MVC, Spring Batch, PostgreSQL, Microservices, Angular (2+), Maven, JUnit.