UNIVERSITY OF GLASGOW – SCHOOL OF COMPUTING SCIENCE
CSC 1009 OBJECT-ORIENTED PROGRAMMING

Group Project (version 1.0)

Warm Suggestion: How to Perform Well?

- Show your instructors how well did you apply the principle and knowledge in the project,
- Do research and understand how the ATM & banking system work, and
- Discuss with the instructors during the tutorial sessions, see if your idea is on the right track.

Background:

Have you ever think about to design an ATM, or a banking system?

According to the Wikipedia, Automated Teller Machine (ATM) or cash machine (in British English) is an electronic telecommunications device that enables customers of financial institutions to perform financial transactions, such as cash withdrawals, deposits, funds transfers, balance inquiries or account information inquiries, at any time and without the need for direct interaction with bank staff.

Thirty years ago, computerized banking system introduced lots of convenience to our daily activities. Even that the black and white console with the original IBM PC Keyboard made in 1981 which only got 83 keys still popular and using in the financial institution nowadays. Why? Because it provides right to the point functionality to the business use.

Intended Learning Outcome (ILO):

1. Appreciate the beauty of CLI application
2. Apply the Java knowledge into the application
3. Experience Internet of Things (IoT), file input/output (I/O), and socket programming (optional)
4. Experiment the software architecture design principle

Tasks:

By the end of the project, you should

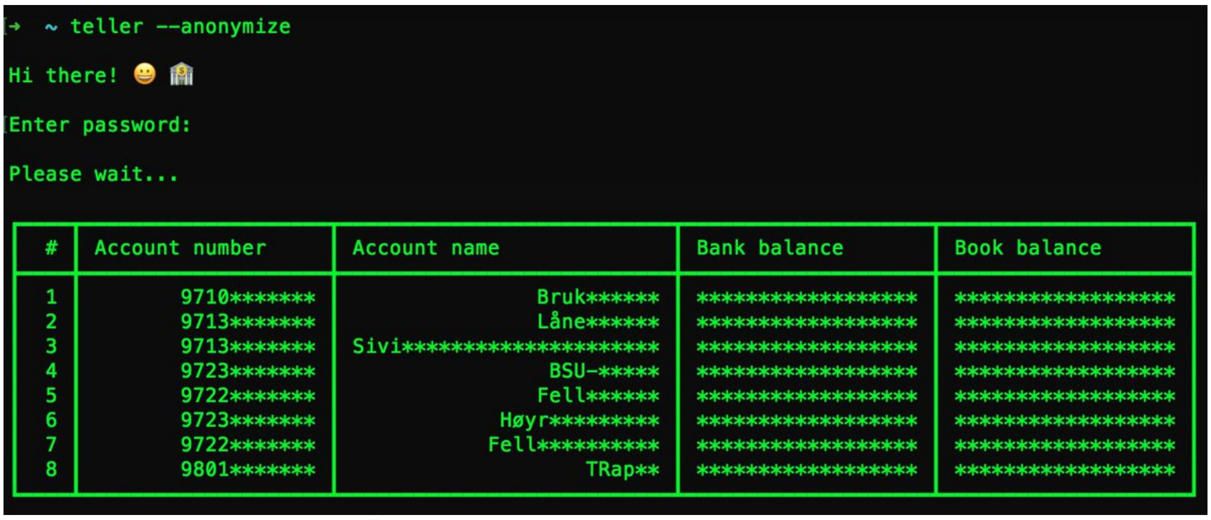
1. Implement an ATM CLI application, with the listed fundamental features below,
2. Brainstorm advanced features with creative ideas,
3. Apply the course contents (e.g. abstract, interface, IoT HTTP call, etc) to the project,
4. Show the necessary best practice in Java, and general coding principle,
5. Write up a report with maximum six pages, which include the sections: features introduction (point form with some explanations), programming highlights (indicate

the file, number line of code and the feature), user interface highlight (some screenshots with description), UML diagram, known issues (e.g. known bugs, limitation, assumption, etc), and
6. A video demonstration, maximum 3 minutes in MP4 format with audio description.

Fundamental Features:

1. Account Information (e.g. how many accounts, account numbers, etc)
2. Balance Check (e.g. remain balance, available balance, etc)
3. Authentication (e.g. password check / reset, etc)
4. Money Transfer(e.g. inter-account transfer, third-party transfer, etc)
5. Settings (e.g. transfer limit, overseas withdraw limit, etc)

Sample Screenshot:



```
→ ~ teller --anonymize

Hi there! 😀 🏦

Enter password:

Please wait...

 ┌───┬──────────────────┬───────────────────────────┬────────────────────┬────────────────────┐
 │ # │ Account number   │ Account name              │ Bank balance       │ Book balance       │
 ├───┼──────────────────┼───────────────────────────┼────────────────────┼────────────────────┤
 │ 1 │      9710******* │                Bruk****** │ ****************** │ ****************** │
 │ 2 │      9713******* │                Låne****** │ ****************** │ ****************** │
 │ 3 │      9713******* │ Sivi********************* │ ****************** │ ****************** │
 │ 4 │      9723******* │                 BSU-***** │ ****************** │ ****************** │
 │ 5 │      9722******* │                Fell****** │ ****************** │ ****************** │
 │ 6 │      9723******* │             Høyr********* │ ****************** │ ****************** │
 │ 7 │      9722******* │            Fell********** │ ****************** │ ****************** │
 │ 8 │      9801******* │                    TRap** │ ****************** │ ****************** │
 └───┴──────────────────┴───────────────────────────┴────────────────────┴────────────────────┘
```

Figure 1: #OpenBanking screenshot by Roy Solberg posted on Twitter

Marking Criteria:

1. Fundamental Features (40%)
    a. 10% Fundamental Features
    b. 10% Architecture Design (e.g., application of OOP)
    c. 20% Usage of Java Programming Language (e.g., usage of various kind of techniques)
2. Advanced Features (15%)
    a. 15% Advanced Features
3. Programming Style and Interface (15%)
    a. 5% Creativity
    b. 5% Coding Style (e.g., comments, indentation, naming)
    c. 5% User Interface Design (e.g., user experience, user interface design)
4. Demonstration (20%)
    a. 10% Report (maximum 6 pages soft-copy PDF only)
    b. 10% Presentation (3 mins video in mp4 format)
5. Peer Evaluation (10%)

a. 10% Peer Evaluation

Schedule:

- Specification release (Wk 5)
- Interim inspection (Wk9)
- Submission (Wk 12)
- Presentation (Wk 13)

Submission:

- No late submission is allowed
- Submission detail will be provided later

What should I do next:

- Start planning the fundamental features with feasible architecture design
- Study, and research related topics from the Internet, and
- Pay attention to the next lecture, more hints will be provided.

=== Happy Programming ===