- Mert
- Aflal
- Yuji
- Anton & Damyan

5CCS2OSD Coursework (due: 30/11/17)

Stage 1 Tasks (Requirements):

- Requirements elicitation (including Ambiguous/incomplete requirements, scope)
- Use Case diagrams (documenting system functionalities)
- Class Diagrams (Classes, Attributes, Visibility, Associations)

Stage 2 Tasks (Design):

- Architecture diagram (whole system)
- Use Case Steps & Use Case Description
- Detailed definition of operations
- Pseudocode for operations/use cases

Stage 3 Tasks (Implementation and Testing):

- Functioning GUI in Java (implementations for operations/classes)
- Code Listing & Test cases with results

Report – put everything together and describe team members roles/contributions to project

Investment Analysis System

An investor may purchase **bonds**.
A bond has the following:

- **term** (number of years to expiry)
- **coupon** (percentage of investment, paid to investor at regular intervals)
- **frequency of payments**
- **name** (E.g. "UK government bond")
- **purchase date**
- **price** (paid on top of investment)

Assume **frequency = 1 per year**, **investment = 100**

Investors **receive back invested sum (100)** at end of term, along with the **final coupon payment**.

- Example; invest £100 in bond b paying 10% coupon annually for 5 years. 4 payments of £10, one of £110.

System enables **new bonds to be defined** and added to **investor's portfolio**.

System should **compute and display the pay-out of all bonds**: sum of payments.

- £150 in the example above.

System should **compute and display bond values**: sum of **discounted** payments, using **inflation rate** r:

X pounds after N years of inflation r is: $\frac{X}{(1+r)^N}$

E.g. £10 in 1 years' time, with 5% inflation, would be worth 10/1.05 = £9.52.

System **computes discount for each payment**, and hence **value of bond**.

- In the example before, a pay-out of £150 at the end of 5 years would only be worth £121.65 for an inflation rate r = 0.05.
- Calculation:
    - £10 in 1 years' time = £9.52
    - £10 in 2 years' time = £9.07
    - £10 in 3 years' time = £8.64
    - £10 in 4 years' time = £8.23
    - £110 in 5 years' time = £86.19
    - Sum of discounted pay-outs = £121.65

For **each bond**, system should **calculate and display** its **Macaulay duration** for rate r:

- Sum of discount payment * corresponding N years
- Sum of the discounted pay-outs

$$\frac{\sum_{p}\left(\frac{p}{(1+r)^N}\right)N}{\sum_{p}\left(\frac{p}{(1+r)^N}\right)}$$

- p is the payment made that year.
- N is the corresponding year to the payment.
- r is the inflation rate.

In example mentioned before, **Macaulay duration** = 517.45 / 121.65 = 4.25 years; r = 0.05.

- Calculation:
    - Sum of discount payment * corresponding N years
        - 9.52
        - 18.14
        - 25.92
        - 32.92
        - 430.95
        - = 517.45
    - Sum of the discounted pay-outs
        - 9.52
        - 9.07
        - 8.64
        - 8.23
        - 86.19
        - = 121.65

For each bond, system should compute and show its **internal rate of return**: the r such that:

$$\sum_{p}\left(\frac{p}{(1+r)^N}\right) = \text{Discounted bond value}$$

In other words, given the output (bond value), the term, and the coupon (percentage), calculate the value of r (inflation rate).

This final use case is quite challenging to implement

Summary of implementation operations/use cases:

- Define new bonds
- Purchase bonds (added to investor's portfolio)
- View Investor's portfolio
- Compute and display the pay-out of all bonds (bond value)
- Compute and display the bond value taking inflation into account
- Calculate and display a bond's Macaulay duration
- Compute and show its internal rate of return

Potential GUI Screens:

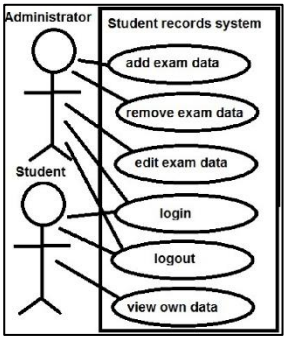
1) Menu
2) Define new bonds
3) Purchase bonds (added to investor's portfolio)
4) View Investor's portfolio
    a. Compute and display the pay-out of all bonds (bond value)
    b. Compute and display the bond value taking inflation into account
    c. Calculate and display a bond's Macaulay duration
5) Compute and show its internal rate of return

Potential Actors:

- Investor
- Admin
- Bonds database

Examples:

Use Case Diagram



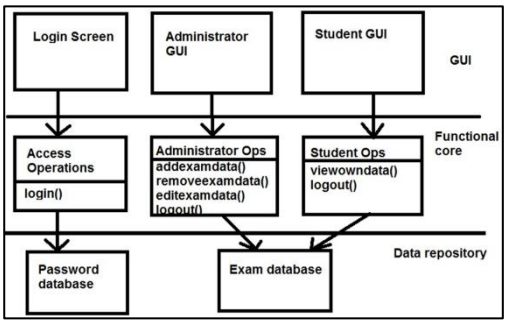
Use Case Description

Use case name	Create a new Blog Account
Related requirements	Requirement A.1
Goal in context	A new or existing author requests a new blog account from the `Administrator`
Preconditions	The system is limited to recognised authors and so the author needs to have appropriate proof of identity
Successful end condition	A new blog account is created for the author
Failed end condition	The application for the new blog is rejected
Primary actors	Administrator
Secondary actors	Author Credentials Database
Trigger	The `Administrator` asks the CMS to create a new blog account

Use Case Steps

Use case name	Create a new Blog Account	
Main Flow	**Step**	**Action**
	1	The `Administrator` asks the system to create a new blog account
	2	The `Administrator` selects an account type
	3	The `Administrator` enters the author's details
	4	The author's details are verified using the `Author Credentials Database`
	5	A new blog account is created
	6	A summary of the new blog account's details are emailed to the author
Extensions	**Step**	**Branching Action**
	4.1	The `Author Credentials Database` does not verify the author's detail
	4.2	The new blog account application is rejected

Architecture Diagram



Class Diagrams