Software Architecture and Design Coursework: Share price technical analysis

February 6, 2024

Coursework Specification

The coursework is worth 60% of the total module mark. It is a group working project and will be delivered in four Sprints. You will also be evaluated for your individual contribution to the project. The marks for the coursework are divided as follows:

Title	Deadline	Marks	Code Review
Sprint 1	14.2.24	20%	15.2.24
Sprint 2	13.3.24	20%	7.3.24
Sprint 3	4.4.24	20%	28.3.24
Sprint 4	29.4.24	20%	25.4.24
Individual Reflection	26.4.24	20%	25.4.24

You will work on the project as a Scrum team. Details on Scrum are provided in your lectures, including an FAQ on how to apply Scrum in the module. It is a teamwork project and therefore it is required to adopt both software Architecture and Design methods and project management methods (Agile, Kanban). You will also need to become familiar with the new tools of a typical software development team and some of the difficult human and technical issues around collaborating on and continuous improvement of online applications.

Building a Share Price Comparison Web Application

Design and develop a robust and scalable **Share Price Comparison** web application using Java. The coursework concerns the charting and comparison of share prices over time (Examples: Yahoo

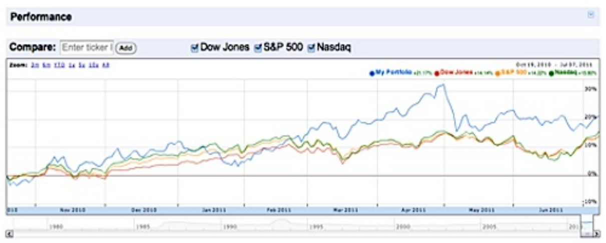

Figure 1: Comparision Chart

Finance, Google Finance and Bloomberg Terminal), applying various architectural concepts including Simple Architecture, Clean Architecture, Compound Components, Basic Styles, Complex Connectors, EIS Styles, and Service-Oriented Architecture (SOA). Your software application should permit users to:

- Obtain daily price information on a given share symbol, between two specified dates, up to a maximum range of two years (a data source such as Yahoo finance can be used.).

- Persistently store such price data on the system, so that the app provides some functionality in the absence of a network connection. Either a relational database (eg., SQLite) or text format (eg., JSON) could be used.
- Display graphs of daily share price over time ranges, for one or two different companies, so that users can view and compare share performance of the companies.
- There is no need to enhance the UI design significantly, the main focus of this coursework is on enhancing the functionality and architectural design.

Deliverables

The coursework will be delivered in 4 'sprints' and it is a group working project. Please pay careful attention to deadlines and essential deliveries for each week. The detailed requirements for each sprint is provided bellow:

Sprint 1: Introduction to Architectural Principles

The aim of this sprint is to check that the project workflow is set-up for the team, requirement and scope if the project are identified. Significantly the key architectural concepts and component specification diagram will be outlined and abstractly implemented in Java.

- Deadline: 14/2/24
- Code Review:15/2/24
- Identify the key requirement for this system and scope of the project.
- Task allocation to each member of the team. Therefore, **GitHub** project for coursework need to be set-up and correct branches should be created. **Project Management Tool** need to be set up. Additionally, **Code of Conduct** need to be defined and agreed within the group.
- Identify architectural concepts encompassing Simple Architecture principles and Outlining a component specification diagram for this system.
- Provide abstract implementation of simple architectural elements for this system in Java.

The following criteria will be assessed for overall quality:

- Requirement Identification: 25 Marks
  - Requirements clearly identified and scoped (5 Marks)
  - Requirements investigated in detail with careful consideration of all aspects (10 Marks)
  - Requirements organized and presented effectively (5 Marks)
  - Requirements aligned to overall goals of project (5 Marks)
- Architectural Design: 30 Marks
  - Architectural concepts well-researched and applied (10 Marks)
  - Component specification diagram clear and complete (10 Marks)
  - Architecture supports requirements and project goals (10 Marks)
- Implementation: 20 Marks
  - Code implements components and meets sprint requirements (10 Marks)
  - Code is clean, organized and commented effectively (5 Marks)
  - Code builds without errors and functions as expected (5 Marks)
- Team management: 25 Marks

    - GitHub project setup complete with branches (5 Marks)
    - Project Management Tool is set up and requirement are added to that (5 marks)
    - Code of conduct defined and agreed upon (5 Marks)
    - Tasks allocated fairly across team members (5 marks)
    - Team worked together on code and commits (5 marks)

Sprint 2: Develop Software Architecture from Requirements

The aim of this sprint is to develop an architecture from a requirements specification. Based on the requirements specification on previous sprint it is required to develop a software architecture using the simple process.

- Deadline: 13.3.24
- Code Review: 14.3.24
- Theory: Developing an architecture from a requirements specification using a simple process.
- The following models need to be created in this phase:
    - Business Concept Model: Create a business concept model from the description.
    - Use Case Model: Create a usecase diagram and define a sequence of steps that need to be performed by an actor using the system to execute the use case and the corresponding system steps
    - System Interfaces: Create system interfaces and operations.
    - Business Type Model: Derive a business type model from your business concept model.
    - Initial System Architecture: Summarise your results through a diagram of a rough system architecture. Discuss how you allocate interfaces to components. In particular, which components implement more than one interface, which only one?
    - Business Interfaces: Use the use cases, your system interfaces, and your first system architecture to discover operations in the business interfaces. Use collaboration diagrams to describe how these operations are used.
    - Considering the principles of Clean Architecture and provide the implementation of components and interfaces.

The following criteria will be assessed for overall quality:

- Architectural Design: 40 Marks
    - Business concept model accurately derived (10 Marks)
    - Use case diagram complete (5 Marks)
    - System interfaces and operations defined (5 Marks)
    - Business type model aligned to concept model (5 Marks)
    - Initial system architecture cohesive (10 Marks)
    - Business interfaces logical and fully defined (5 Marks)
- Implementation: 30 Marks
    - Code implements clean architecture principles (15 Marks)
    - Components and interfaces implemented as designed (15 Marks)
- Team management: 30 Marks
    - Team coordination apparent in project management tool (15 Marks)
    - GitHub updates (15 Marks)

Sprint 3: Implementing Compound Components and apply appropriate domain Independent styles

The aim of this sprint is to develop compound component for this web application and apply appropriate domain Independent styles to the system to make it more reusable and scalable.

1. Deadline: 4.1.24
2. Code Review: 28.3.24
3. Divide the architecture into compound components and apply basic styles for creating reusable and scalable UI components. Next, implement these components.
4. Identify and implement the most appropriate domain Independent styles you learnt through out this module (Pipes and filters, Adapter, 2-tiered, N-tiered, Layered, Blackboard, Model-View-Controller).

The following criteria will be assessed for overall quality:

- Architectural Design: 40 marks
  - Architecture divided into logical compound components (15 marks)
  - Appropriate styles selected and applied (15 marks)
  - Design supports reusability and scalability (10 marks)
- Implementation: 30 Marks
  - Compound components implemented correctly (15 Marks).
  - Component and interfaces implemented as designed (15 Marks).
- Team management: 30 Marks
  - Team coordination apparent in project management tool(15 Marks)
  - GitHub updates (15 Marks)

Phase 4: Apply EIS Enterprise Styles and Service Oriented Architecture

- Deadline: 29.4.24
- Code Review: 25.4.24
- Redesign and improve your system architecture by applying the most appropriate EIS enterprise styles.
- Implement the Java EE components of the system
- Introduce the principles of SOA emphasizing service modularity and interoperability.
- Use suitable test cases to test your completed system, and document the test cases and results in your report.

The following criteria will be assessed for overall quality:

- Architectural Design: 25 Marks
  - Appropriate EIS styles selected and applied (10 Marks)
  - SOA principles applied effectively (10 Marks)
  - Design is modular, interoperable, and scalable (10 Marks)
- Implementation: 25 Marks
  - EIS components implemented correctly (10 Marks)

  - SOA implemented correctly (10 Marks)
  - Code complete, functioning, and meeting requirements (5 marks)

- Team management: 25 Marks
  - Team coordination apparent in project management Tool (10 Marks)
  - GitHub updates (15 Marks)
- Group Report: 25 Marks
  - Report complete and well-structured (10 Marks)
  - Design rationale explained clearly (10 Marks)
  - Code and other files submitted correctly (5 Marks)

Code Review Meeting

Each group will undertake four graded code reviews:

1. Sprint 1: 15.2.24
2. Sprint 2: 14.3.24
3. Sprint 3: 28.3.24
4. Sprint 4: 24.3.24

The meetings will take place during the lab sessions. Each group will be given 15 minutes maximum for the code review. Your group will be allocated a time for the code review. The details of the individual review points are below. These meetings must be attended at the stated time. Guidelines for grading the group:

- Being late for the meeting or not being ready when the meeting starts will result in the grade for that review being capped at 40%
- Not attending the meeting will mean the code review will be marked at 0%
- All team members should attend the code review, however commitments and other considerations will be taken into account. Individuals attendance at reviews will be monitored to ensure the team is contributing collectively to the project.
- Being ready means that you are ready to present the points for the code review. This means that you have a computer with the various tools logged into (e.g., GitHub, Travis CI, etc.) and a building version of the application in IntelliJ.

Group Project Report

- Submit a report named with the member of your team
- Include all the design of the system and explain the rationale for those.
- Include a list of the project team meetings identifying time/date, who attended, what was discussed and what decisions were made.
- Include test case definitions and test case results.
- The report should be submitted as a pdf format.
- The source code and test data files should be submitted as a zip file.

Individual Reflection Report

Your individual reflection report should contain the following sections:

1. Introduction
   - Short description of the project
2. Design process
   - What design techniques did you use?
   - Which ones did you find the most useful for yourself (a) as an individual and (b) for the group
3. Development
   - What did you personally contribute to the project?
   - How happy were you with your own contribution?
   - Is there anything you would have liked to have done more of?
   - What did you find easy?
   - What did you find difficult?
   - What would you do differently next time?
4. Teamwork
   - How well do you think you worked as a team?
   - What did you learn about teamwork from this project?
   - What percentage of the overall work for the project would you estimate that you personally did?
5. Reflection
   - Lesson learnt
   - What would you like to do more of following this project?

The following criteria will be assessed for overall quality:

- Individual Reflection: 20% of total coursework grade
  - Accurately describes contributions to project (5 Marks)
  - Thoughtful self-assessment and reflection (10 Marks)
  - dentifies lessons learned and improvements (5 Marks)