F21AS Advanced Software Engineering: **Group 04 / Stage 1**

Criteria	Weight	A (70-100%)	B (60-69%)	C (50-59%)	D (40-49%)	E/F (<40%)	Marks
Development plan and design decisions	25%	Clear, well thought out, well justified, and submitted by the deadline	Mostly clear, thought out and justified, submitted by the deadline	Some issues with clarity, planning or justification, but submitted by the deadline	Significant issues with planning or justification, or not submitted on time	No real indication of planning or thinking, or not submitted	15
Functionality	30%	Functional requirements have been met	Some requirements not fully met	A number of requirements are incomplete	Significant limitations in functionality	Very little achieved	14
Implementation and coding	25%	Good OOP design, clear modular well commented code, clear class diagrams, appropriate use of version control	Generally good OOP design and coding with readable class diagrams, appropriate use of version control	Some issues with OOP design or coding, class diagrams lack clarity, limited use of version control	Significant issues with OOP design and coding, class diagrams poorly presented or absent, poor use of version control	Poor OOP design and coding, incomprehensible class diagrams, no use of version control	14
Exception handling and testing	20%	Effective use of exceptions, thorough unit testing and test data	Generally good use of exceptions and unit testing with sensible test data	Some issues with the use of exceptions, unit testing and test data	Significant limitations with exception handling and testing	No useful exception handling and testing	12
						Total (%)	55

Feedback:

The development plan covers key areas including class diagram, project plan and data structures. More justifications and details would have been useful. The use of red-black-tree is not justified. Sequence Diagram have some accompanying narrative text.

A number of functional requirements are incomplete. The GUI is not usable, I couldn't see Items, or create Order.

The code is mostly undocumented. The code tends towards a procedural structure rather than an OOP structure. Good use of version control with clear branching for team members, some documented commits, documentation on code merge could be more detailed. Try-Catch block should contain only the code that could fail with the caught exception, usage is too wide, potential danger with nested exceptions. Uncaught exceptions. Errors running the program.

Junit test scripts have been written for a few methods; more tests are needed. As required, you have implemented exception and error handling, and a custom exception classes and exceptions are thrown in the constructor of some classes. Should use assertAll for test with multiple assertions.