MoSCoW Prioritisation

MUST HAVE

- Code fully integrated into a Version Control System (Git).
- A build of the application present in the root folder.
- A working application that is interactive with a front-end command-line interface (CLI).
- A fat .jar which can be deployed from the command-line.
- A completed project management board; user stories, acceptance criteria + story points estimation.
- A risk assessment outlining risks during the project timeframe (pdf).
- A 'back-end', following best practices and design.
- An application build, including possible dependencies, produced with integrated build tool (Maven).
- Unit tests for validation of the application. (Junit & Mockito)
- The project connecting via JDBC to a local-based MySQL instance
- A CRUD functionality following the Enterprise Architecture Model; customers, items, and orders entities.
- add/delete an item to and from an order.
- A relational database used to persist data for the project
- A primary key in each of the entities (tables)
- A sensible package structure.
- A completed README.md, explaining how to use and test the application.
- At least one ERD and one UML diagram (png)
- A copy of the presentation (pdf)

SHOULD HAVE

- The Git repository utilising the feature-branch model: main/dev/multiple features.
- At least 5-7 risks within the risk assessment.
- Relationships between tables modelled using an ERD.
- Before-and-after ERD's and UML.
- Unit test coverage of CRUD functionality of the src/main/java folder, aiming for 80%.
- An adherence to best practice (OOP principles, SOLID, refactoring)
- A working ".ignore" for ignoring build-generated files and folders
- The means to calculate the total cost of an order

COULD HAVE

- The project connecting via JDBC to a GCP-based MySQL instance.
- Test coverage of CRUD Functionality above 80%.
- A 'Date placed' field in the Orders table with timestamps.
- A description and a stock count in the Items table.
- An evolution of ERD/UML Diagrams over time.
- A User-system; with a username and password.

WON'T HAVE

- Extra functionality outside of the specification.
- Unnecessary imports / dependencies / database tables.
- A Graphical User Interface.
- Any incomplete documentation.