Sprint plan

Goal for Milestone #2 (End of Week 10: 16/5):

* Fully functional with client – server – database interaction:
  + Client can sends requests to server to execute on database
    - Admin (Create / Update / Delete for each category)
    - User (Buy, Sell, See current transaction)
  + Implement Socket & Serialisation *(Rodo)*
  + Switch based on Request -> returns response (*Daniel*)
  + Matching buy & sell orders to finish them:
    - Buy & Sell orders on the same asset with acceptable prices can be executed
    - Prioritising orders
* Improve GUI:
  + Finalise interface for account type “user”
  + Close program button
  + Improve Design to High Fidelity
* Unit Testing & Further Implementation Plan:
  + Client (Controllers – Not GUI)
    - Socket
    - Custom Exception Classes
  + Common
    - Test initiation of data classes
    - Data Boundaries
  + Server Classes
    - DataSources
    - SocketServer
* Multi-threading (challenge – only attempt):
  + Server can interact with many clients at once (one thread per Socket connection)
  + Client can listen for notification from server

Dan

* Design, Documentation & Implementation: Matching ‘buy’ & ‘sell’ orders (try to match a new order with all current orders, based on asset type & prioritise lower price)
* Black Box Unit Testing: SocketServer & SocketClient

Daniel

* Design, Documentation & Implementation: Switch (Analyse request structure – Query & Update)
  + Query (Sender, String “whatever”) -> Server queries and return Response(true / false, IData)
  + Update (Sender, String “add / update” / “delete”, IData) -> Server updates database based on IData and returns Response(true / false, postUpdateIData)
* Black Box Unit Testing: GUI Controllers

Lyn

* Design, Documentation & Implementation: GUI (User) + Exit button + Design to see current BUY orders.
* Black Box Unit Testing: DataSouces & Features (Login System & Hash Password)

Rodo

* Design, Documentation & Implementation: [Socket & Serialisation](https://www.codeproject.com/Tips/991180/Java-Sockets-and-Serialization) (Establish connection between client & server. Client sends request to outputStream -> Server reads from inputStream -> Server writes response to outputStream -> client reads response from inputStream)
* Black Box Unit Testing: Common Classes (IData + Request / Response)