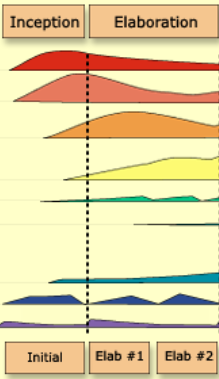
SPRINTS  
  
  
    Sprint 1. From 11/10/2018-> 01/11/2018.  
  
Backlog (higher up the list means higher priority):

|  |  |
| --- | --- |
| Implement client-server connection architecture (RMI Design Pattern) | 1 hour |
| Implement Core User Commands | 1 hour |
| Implement Observer Design Pattern | 2 hours |
| Test the implemented design patterns (white box) | 10 hours |
| Improve upon UML diagrams during Tests | 10 hours |

### First day 11/10

Implement MVC (Model View Controller) Design Pattern on Client and on Server

Implemented client-server connection architecture. (Remote Method Invocation)

Implement observer Design Pattern

Second day 25/10  
Implement Adapter Design Pattern.

Implement Core User Commands.

### Third day 01/11

Improved upon UML diagrams as we were adjusting previously implemented design patterns.

Sprint retrospective (general thoughts)

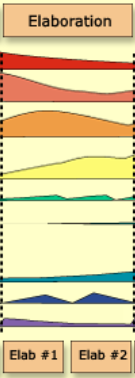
We did not possess the knowledge to fully accomplish the core tasks. Reason: the group was busy studying design patterns that must be used to implement the required features.  
Java architecture must be implemented first as more knowledge is available to group members in that direction.  
The UML must be adjusted and prepared for improving what has already been done in the first sprint.

We need a more accurate backlog for the next sprint.

Sprint 2. From 08/11/2018 -> 22/11/2018  
  
Backlog (higher up the list means higher priority):

|  |  |
| --- | --- |
| Improve all the Design Pattern implementations from the first Sprint up to a functional level. | 24h |
| Implement the Singleton design pattern where object instance must remain unique. | 1h |
| Test the improved design patterns (white box) | 3h |
| Improve upon UML diagrams during Tests | 3h |

### First day 08/11

Optimized the RMI implementation.  
Improved the UML and design patterns for certain classes.

### Second day 15/11

Implemented the Singleton Design Pattern.

Changed the functionality of the Model Manager class for adapting the user requests to the database.

### Third day 22/11

Improved the database connection between SQL and Java.

Edited the UML according to the improved design patterns.

### Sprint retrospective (general thoughts)

Design pattern implementation has again been underestimated even if we have tried to be more accurate with the backlog definition. Therefore, we will dedicate even more time towards the correct implementation of design patterns the next sprint as these are the core of our application functionality.

The goal is to have 95% of the required design patterns implemented by the end of the next Sprint.

Sprint 3. From 29/11/2018 -> 07/12/2018.  
  
Backlog (higher up the list means higher priority):

|  |  |
| --- | --- |
| Improve all the Design Pattern implementations from the second Sprint | 15h |
| Start Normalizing and implementing core parts of the database. | 5h |
| Test the improved design patterns (white box) | 10h |
| Improve upon UML diagrams during Tests | 5h |

### First day 29/11

Ensure that Java SQL library is used where required.

Ensure that all Model Objects are serializable

### Second day 06/12

Improve upon Design Pattern implementations

Tested the improvements

Adjusted UML diagrams accordingly  
  
Third day 07/12  
Started database normalization of SQL tables.

Improved data integrity & accessibility

Reduced data redundancy.

Started building SQL database from EXCEL.

### Sprint retrospective

Excel tables are now more informative for users and ready for SQL integration.

The MVC design pattern helped us achieve separation of concerns and a satisfactory level of Logical Architecture.

This allowed for applying the concept of designing in layers (low level layers for general services and high-level layers that are application specific)

* e.g. low-level layers – client server communication
* e.g. high-level layers – mediator package (acts as an adaptor for the client server communication)

The Singleton pattern helped us restrict the access towards an object’s instantiation and ensured that data is always flowing through one and the same instance of an object.

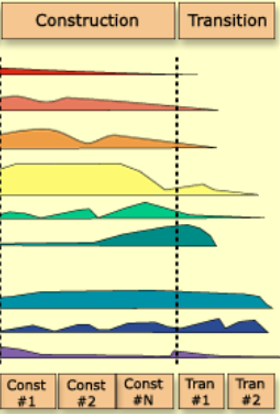
The observer design pattern will allow us to notify the client about any change in the database.

Specifically, it notifies the client about changes in the data that it is currently viewing and forces the client to update that data from the server.

The Dependency inversion principle allowed us to separate the dependency between higher and lower level application layers by creating interfaces for communication between them.

Sprint 4. From 10/12/2018 -> 12/12/2018  
Backlog (higher up the list means higher priority):

|  |  |
| --- | --- |
| Database core tables normalization must be remade. | 16h |
| Discuss with client about current attribute domains. | 2h |
| Insert available data into finalized SQL tables. | 8h |
| Improve Java mediator package classes. | 4h |
| Implement Java Object to SQL Data conversion. | 5h |
| Decide on database attribute design. | 1h |

First day 10/12  
Database normalization continued.

Renamed attributes, item descriptions, cost descriptions (with the owner’s consent) in order to improve readability of the information.

Added primary keys for item costs and serial Id and category as proper attributes.

Added triggers where data is dynamically modified.

Second day 11/12  
Populated the database with the information (users, inventory, financial information).

Database connected from SQL to Java, implementing the attributes for financial and inventory information.

### Third day 12/12

Continued populating the database with information.

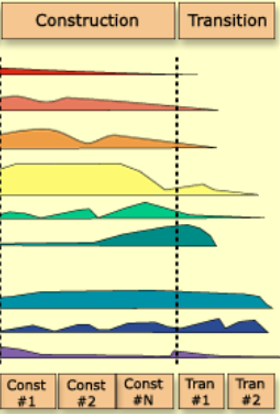
Continued improving the java mediator package classes.  
  
Sprint retrospective

Everything is now ready for starting the white box testing phase of the core application functionality without an Interface. This must be accomplished by the end of the next Sprint. Afterwards, this iterative process must be applied to all user commands in terms of testing. NOTE: Unit testing cannot yet begin.

Sprint 5. From 13/12/2018 -> 17/12/2018  
Backlog:

|  |  |
| --- | --- |
| Continue populating the database with information | 2hours |
| Continue improving the java mediator package. | 5hours |
| Start the white box testing faze on all use case scenarios | 5hours |
| Start checking on the correctness of documentation for the entire development process | 16hours |

First day13/12

Continued populating the database with information.

Continued improving the java mediator package classes.

All the above while white box testing the application functionality by running through the use cases.  
  
Second day 14/12

Continued improving the java mediator package classes.

Populated the database with the information (users, inventory, financial information).

All the above while white box testing the application functionality by running through the use cases.

Third day 17/12

Checking if the entire development process has been well documented and adjusting reports for a understanding by the regular user.

Sprint 6. From 18/12/2018 -> 19/12/2018.

Backlog:

|  |  |
| --- | --- |
| Continue checking the correctness of documentation for the entire development process | 20h |

First part 18/12

Continued checking the correctness of documentation for the entire development process  
  
Second part 19/12

Continue checking the correctness of documentation for the entire development process

Submitted all required documentation and current software version.