BattleShips Project by The Best Team

Team members:

David Ye(91031611), Thomas Dalzell (91031633), and James Fearnley (91031576)

IT5014v4d\_Group\_Project

**Project Report:** Battleships By; The Best Team (David Ye, Thomas Dalzell, James Fearnley)

1. **Planning:**

* **Planning**

During the first meeting we decided on the users stories in order to help us select the features that we were going to include in our program. We also designed the initial UML diagram to help us understand how these features were going to fit together and how the inheritance of the different classes might work. We also created a diagram (IMG-1409.JPG) of how the program was going to function. This diagram helped us visualize how the program was going to run as well as making sure the team were all on the same page.

* **Agile approach:** We chose to use Scrum as an iterative approach toward the project development. Many parts of the code and planning went through iterations including pseudocode, first python code, integrated code, and final functional code, with testing in between the stages. This was us implementing the agile scrum methodology using iterations and coming together for sprint meetings to make sure we had fulfilled each task and if not what was required to get that part to an acceptable level.

* **Task delegation and collaboration** emails (as seen in appendix)

Tasks were created in trello as ‘to do list items’ and each team member voluntarily picked up items and items without ownership were delegated. The team faced some challenges with members facing personal challenges during the project. Also the use of trello as an organization board caused some issues as initially (in the first sprint meeting) we did not immediately place names making delegation a bit ambiguous. The result of this was that delegation of the tasks took time to sort out properly but ultimately delegation was implemented. By subsequent sprint meetings delegation was sorted out effectively with everyone picking the parts of the code that they were able to do best with their knowledge and abilities taken into account. For some task we had multiple people working on the same task to allow us to bring bits of code together to achieve the best outcome (as seen in the Burndown\_overall.xls J,D,T). We would then choose code which was the most functional (the best code) and use this to standardise our files so that when parts of the program were brought together (new iterations), there would be little work involved in getting our program to function. An example of this would be the board. We all developed one then gave each other feedback on the functionality and aesthetics and decided that Davids version was the best suited for our needs while also implementing parts from other versions. When someone was stuck on one bit of code they could incorporate something written by another person. An example of this was the validation code where for the inputs of the ship placement, Tom and David had both developed conditional statement validations for these inputs. As a team we decided that Tom’s conditional statements would be best suited and that Davids import and function codes were used were Tom had not been able to figure a part out. This approach lead to us being able to quickly overcome many challenges within the project often with all of us taking turns and asking each other advice on how to troubleshoot issues.

* **Sprint minutes and meetings** (2 day sprints as on trello and written meeting notes in appendix)

We had sprint meeting every 2 days (except for a weekend which was 4) to plan for the next few days (see IMG-1416.JPG for example or any of the files in the sprint meeting folder) . We would then take a look at the completed work on the due date and decide if work on it need to be continued or if it was completed. We would then see how these interacted with other existing code and see whether or not they still meet our expectations, fulfill user stories, ad meet our other project requirements. During these meetings we were able to review features we had planned to implement and decide if they were possible or no longer possible with the time we had left, such as with planning to implement AI for an opponent. These task were placed on trello with a due date 2 days after the sprint meetings. There is evidence of these in the appendix as well as proof by delegation and creation of tasks in trello as many of these tasks were created at the sprint meetings.

After these meetings we would then spend time collaborating on codes in order to help team members that were facing challenges in code that they had been tasked with. This collaboration time allowed us to help each other with the practical reasoning of our code and understand how our code is going to work.

1. **Analysis and design**

* **User stories:**

We chose a number of user stories to implement into the project. This was during a sprint session on the 29/11/2018 at 4:00. We decided to aim to implement a high number of user stories (16 chosen and 9 included in final code) as shown below:

User stories:

- welcome message (**included** at beginning of game)

- game begin and game over message (**included** but not implemented)

- rules message (**included** as how to place ships)

- game controls message (**included** in how to place ships)

- placing ships one by one (**included**)

- only placing ships inside the board (**included**)

- not placing ships on top of other ships (not included)

- messages ‘hit and miss’ (for human and cpu) (**included** although went for player vs player)

- messages for you already tried that coordinate (**included**)

- message ship sunk (**included** but not implemented)

- play again message (**included** but not implemented)

Optional user stories:

- giving the player a name (not included)

- board visually appealing (**included**)

- keeping track of turn count for a score (not included)

- having a ship of a different length (not included)

- computer talk back ("ahh ya got me", or "haha take that") (not included)

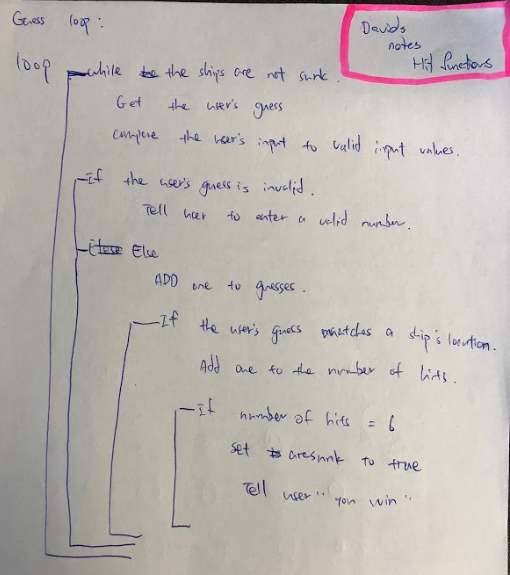
* **Design notes** – uml paper design notes

We have many design notes from throughout the course of this project. We have some initial UML designs and a final UML design which we designed using Microsoft Visio can be located in the appendix as BattleshipsUML.vdsx and BattleshipsUMLFinal.vdsx. The final UML deviated from the initial UML as we found new names for classes and removed classes which we did not end up implementing at all. The main file inherits all classes, and the only other class to inherit another class is guess (which inherits board).

The design and planning notes can be found in the appendix in the folder “design and planning notes” there are notes from each of the team members for the main parts of the program we worked on. We made these to help us each clarify how the conditional statements would work as well as the order the program would run in from stat to end. Once we had the design notes we created pseudo code for those functions, then began creating the python code. After that the team integrated the pieces of code into a mian final file and we managed to get all of the priority user stories implemented successfully with one or two pieces commented out as we ran out of time to integrate them properly. Many parts of the code went through iterations from planning pseudocode to first python code, to integrated code, to final code, with testing in between the stages. This was us implementing the agile scrum methodology.

The project outline did not require us to have a fully functioning game so we knew this was an ok point to cease code development and move onto writing the report.

An examples of this is;



These design notes show how the loop would work for the guess class’s hit functions. (Davids note)

ship placement design (James email)

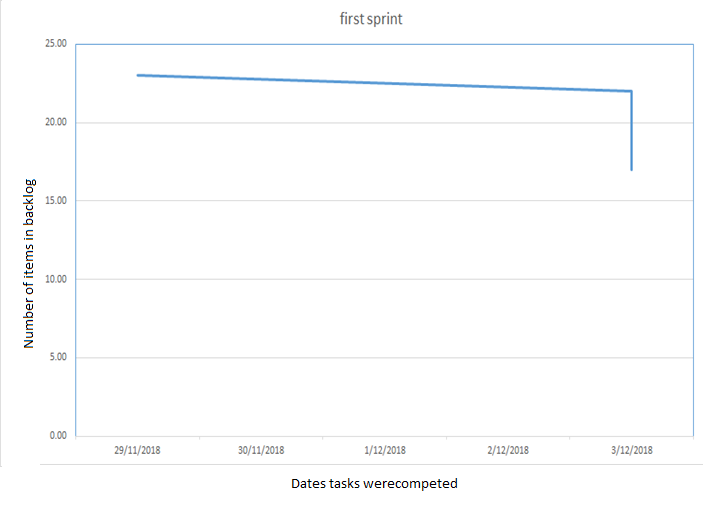
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| 0 | HV | HV | HV | V | V |
| 1 | HV | HV | HV | V | V |
| 2 | HV | HV | HV | V | V |
| 3 | H | H | H |  |  |
| 4 | H | H | H |  |  |

* **Pseudo code** – We have three examples of pseudo-code, we have initial pseudo code from before we had decided if we wanted two human players or a human and computer player. We also have pseudo code from our initial meeting email. We used this pseudo code to design our ship placement and ship hit or miss functions for the product. These initial files can be found in the appendix as (placement), Sudo\_code\_initial2 (hit or miss).doc, the final can be found as Sudo\_code\_battleships.doc (placement and hit or miss)
* **Product backlog** - the product backlog and the timeframes in which items moved from conception to completion can be viewed easily in the excel sheet Burndown\_overal.xls Where by 23 items were developed and implemented into the product with almost all of the items working on the final day of coding. No item was left in the backlog for too long as we endeavoured to clear the main tasks before each sprint. Any items still to be completed gained priority and was quickly done. Items were added to the backlog at each sprint meeting as we decided what parts of the product were to be created next. Some items required a follow up iteration and were added again. Reasons for requiring a second iteration were to implement new user stories to existing code, to integrate existing code to other code sometimes rework of the existing code was required, and also to tidy up the code the best we could to fit inline with best practise guidelines in the python language. Product backlog can also be seen to decrease over different time periods in the burndown charts in the implementation section of the report.

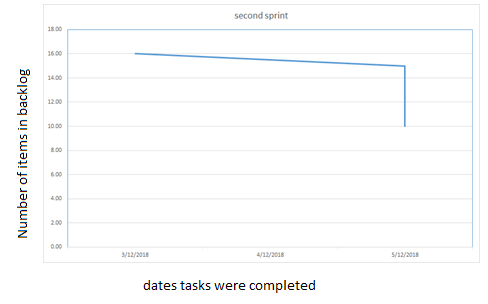
**3. Implementation**

* **Trello screenshot**s showing what we implemented for each sprint: these can be found in the appendix in the file “sprint meeting” and “trello screenshots” for evidence. A breakdown of the creation and completion of tasks can be found as Burndown\_overall.xls.
* **Burndown charts** for each sprint and one for the overall project larger files and excel files can be located in the appendix.

sprint one burndown chart: 29th



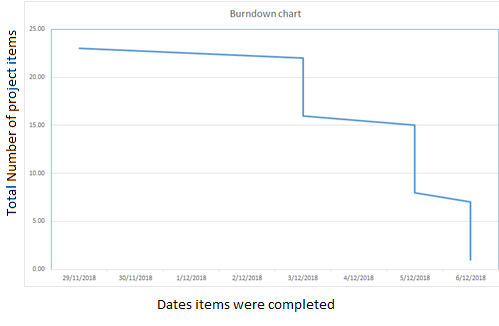
sprint two burndown: meeting on the 3rd and 4th



sprint three burndown: meeting on the 5th and 6th



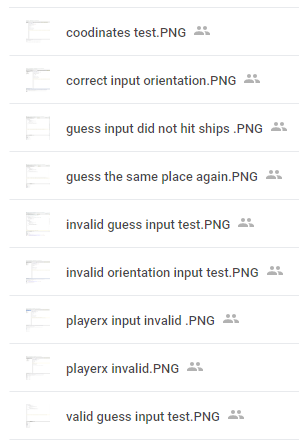
overall project burndown chart: days 29th-6th



1. **Testing**

* Testing files used to test the classes and functions

Testing codes and screenshots are in the testing screenshot files stored in Appendix.



This is a list of the testing files that we have as evidence of testing. We conducted testing at every step, during the creation of functions, classes, and integration of code for the final product.

1. **Appendix**

* Photos and emails all evidence stored in here to be referenced.

<https://drive.google.com/open?id=1jS6-Z5Jx7Hl25oF6Wz7zAHISRnQt-W3v>

<https://drive.google.com/open?id=1NDDOLcqXLX-1beeSQItQwE5oLzrPWZ5C>