**Sprint 1**

**Goals:**

* We should have game mechanics code fully implemented with a playable board
* An AI engine must exist using randomly-picked moves
* Server code should be implemented allowing a user to connect to the open port and play a game against the AI. Difficulty level and AI-AI should be implemented int he protocol, however it is not necessary to have the related functionality implemented yet

**Backlog**

| **Task** | **Time Finished** |
| --- | --- |
| Create Game class | 1 |
| Create State class | 1 |
| Create Board class | 1 |
| Parser: Take input | 2 |
| Parser: Split command on delimiters | 2 |
| Parser: Check command for validity | 3 |
| Parser: Direct command to function | 5 |
| Print out board | 3 |
| Undo function | 7 |
| Create left operator | 4 |
| Create right operator | 4 |
| Create forward operator | 4 |
| Check operator validity | 5 |
| Perform move | 6 |
| Termination condition check | 9 |
| Report game results | 5 |
| Open port on server | 9 |
| Read input | 5 |
| Check for password authentication | 8 |
| Send output | 3 |
| Find possible moves for a piece | 4 |
| Random Algo: Check obvious moves | 8 |
| Random Algo: Pick random move | 8 |
| Set base structure for difficulty | 7 |

Nathan is working on Game Mechanics

**Game Mechanics**

* Create Game class to handle a game
* Create State class
* Create Board class

Jonathan is working on Parser

* Parser
  + Take input command
  + Split command on delimiters
  + Check command for validity
  + Direct command to appropriate function if valid
* Print out board
* Undo function(revert to previous state stores in vector of States
* Operators
  + Create left operator
  + Create right operator
  + Create forward operator
  + Check operator validity
* Perform move

Victor is working on Termination Condition Check

* Termination condition check (is the game over?)
* Report game results

Ryan is working on Game Server

**Game Server**

* Open port on server
* Read input
* Check for password authentication
* Send output

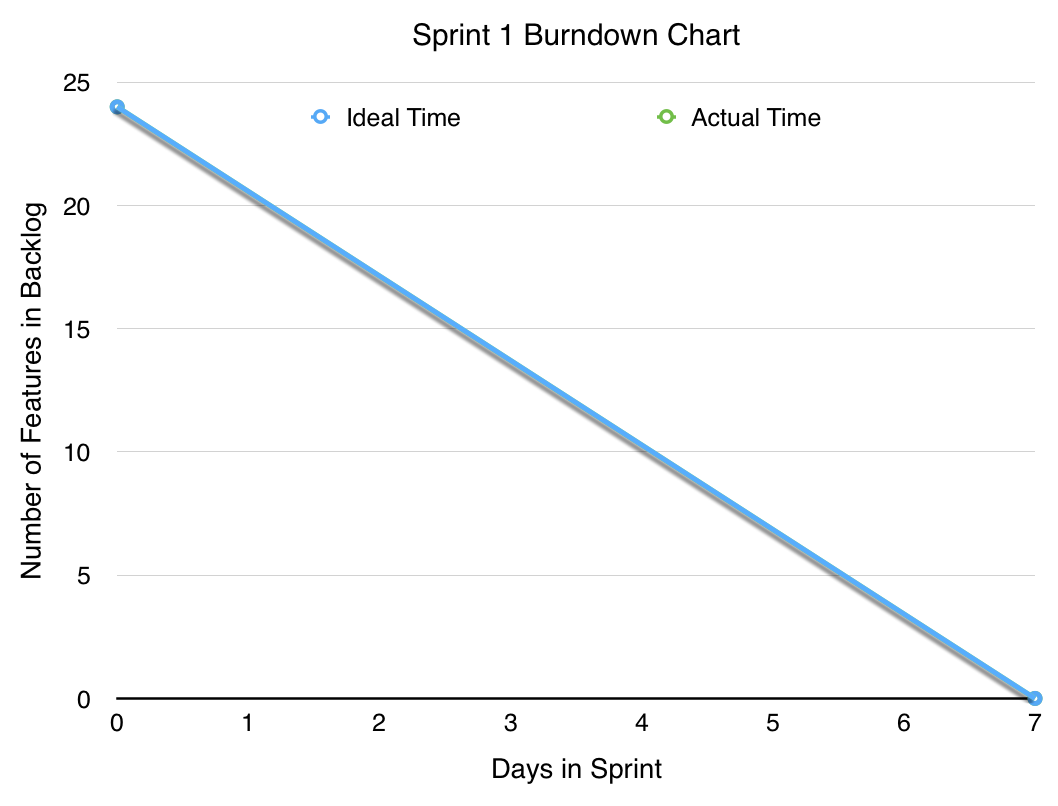
Ryan is working on AI Engine framework(Random algorithm implementation now)

**AI Engine**

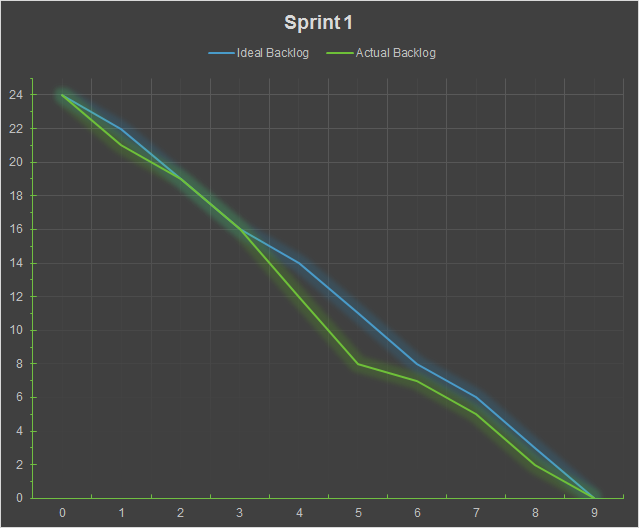
* Find possible moves for a piece
* Random Algorithm
  + Check for obvious moves to make(if a piece can be overtaken)
  + If no obvious moves to make, pick one at random
* Setup base structure for difficulty(assign all difficulties to random algorithm for the time being)

**Burndown Chart**

First Deliverable:



Second Deliverable:



| **Time Spent** | **Ideal Backlog** | **Actual Backlog** |
| --- | --- | --- |
| 0 | 24 | 24 |
| 1 | 22 | 21 |
| 2 | 19 | 19 |
| 3 | 16 | 16 |
| 4 | 14 | 12 |
| 5 | 11 | 8 |
| 6 | 8 | 7 |
| 7 | 6 | 5 |
| 8 | 3 | 2 |
| 9 | 0 | 0 |

**Sprint 2**

**Goals:**

* AI engine should implement MIN-MAX algorithm

Nathan will be working on the Min-Max Algorithm

* AI engine should implement Alpha-Beta Pruning

Ryan will be working on the Alpha-Beta Pruning Algorithm

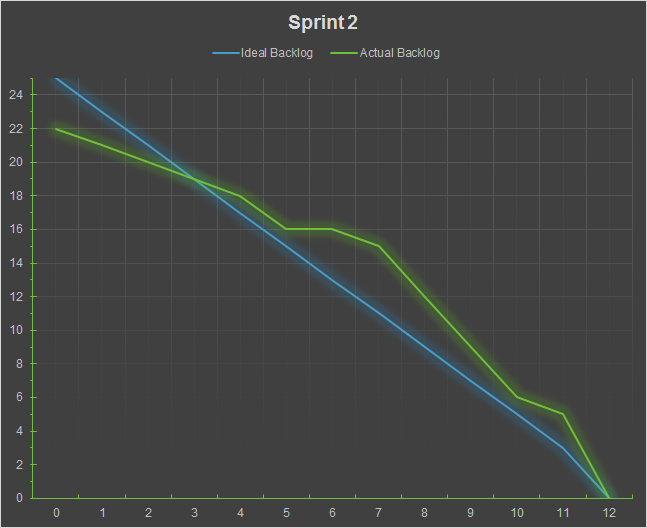
* Design and implement a cut-off state(depth limit)

Jonathan will be working on the Evaluation Function and Tree/Node classes

Victor will be helping where needed

**Backlog**

* MIN-MAX Algorithm
* Update Medium difficulty to use MIN-MAX algorithm
* Alpha-Beta Pruning
* Update Hard difficulty to use Alpha-Beta Pruning



| **Task** | **Time Finished** |
| --- | --- |
| Create Min-Max Algorithm function | 0 |
| Create Alpha-Beta pruning function | 1 |
| Create Evaluation Function | 0 |
| Min-Max: Evaluation of each piece | 9 |
| Min-Max: Assign Value of board | 5 |
| Min-Max: Returning Max/Min value | 8 |
| Evaluation Function: Create Tree for moves | 4 |
| Evaluation Function: Rank child nodes | 5 |
| Alpha-Beta: Prune Min-Max | 11 |
| Aplha-Beta: make more effiecent | 12 |
| Execute Min-Max Algorithm | 9 |
| Execute Alpha-Beta pruning | 12 |
| Execute Evaluation Function | 8 |
| Fix Display-to-screen errors | 2 |
| Implement Human-AI: Min-Max | 10 |
| Implement Human-AI: Aplha-Beta | 12 |
| Implement Human-AI: Evaluation Function | 9 |
| Implement AI-AI in general | 7 |
| Fix server-client AI-AI issues | 3 |
| Implement AI-AI: Min-Max | 10 |
| Implement AI-AI: Aplha-Beta | 12 |
| Implement AI-AI: Evaluation Function | 8 |
| Implement AI Difficulty: Easy | 0 |
| Implement AI Difficulty: Medium | 10 |
| Implement AI Difficulty: Hard | 12 |

| **Time Spent** | **Ideal Backlog** | **Actual Backlog** |
| --- | --- | --- |
| 0 | 25 | 22 |
| 1 | 23 | 21 |
| 2 | 21 | 20 |
| 3 | 19 | 19 |
| 4 | 17 | 18 |
| 5 | 15 | 16 |
| 6 | 13 | 16 |
| 7 | 11 | 15 |
| 8 | 9 | 12 |
| 9 | 7 | 9 |
| 10 | 5 | 6 |
| 11 | 3 | 5 |
| 12 | 0 | 0 |

**Sprint 3**

**Goals:**

* Updated version of design document
* Implemented GUI client
* Post production notes
  + Changes made to the design and why, difficulties, solutions, lessons learned
  + Individual workload percentages with brief description of each user's contribution
  + Development log

**Backlog**

**GUI client**

* Connection to server
* Send move to server
* Interpret server responses
* Display board

| **Task** | **Time Finished** |
| --- | --- |
| Update Board-Game Layout | 4 |
| Perfect Command Transcribing | 2 |
| Creation of GUI Class | 3 |
| GUI Client: Added functionality to buttons | 2 |
| GUI Client: Connecting to server | 3 |
| GUI Client: Sending move to server | 2 |
| GUI Client: Interpret server responses | 2 |
| GUI Client: Display Board | 1 |
| GUI Client: Fixed row and columns | 1 |
| GUI Client: Testing user access | 1 |
| GUI Client: Added images | 1 |
| GUI Client: Added buttons | 1 |
| GUI Client: Update Color Scheme | 1 |
| GUI Client: Improve efficiency | 3 |
| GUI Client: Fix functionality of buttons | 1 |
| GUI Client: Update connection issues | 4 |
| GUI Client: Password Screen | 2 |
| GUI Client: Centering client | 2 |
| GUI Client: Connection functionality | 2 |
| GUI Client: Allow insensitive password | 1 |

| **Time Spent** | **Ideal Backlog** | **Actual Backlog** |
| --- | --- | --- |
| 0 | 20 | 20 |
| 1 | 16 | 15 |
| 2 | 12 | 11 |
| 3 | 8 | 9 |
| 4 | 4 | 2 |
| 5 | 0 | 0 |