# Blackjack II

# 

Prepared by:   
Brandon Escota, Max Nelson, Wilson Pena

11/01/17

Contents

[Project Title 1](#_Toc494890065)

[I. Overview 3](#_Toc494890066)

[II. Team Members 3](#_Toc494890067)

[III. Deliverables 3](#_Toc494890068)

[IV. Features 3](#_Toc494890069)

[V. Application Wireframes 3](#_Toc494890070)

# Overview

Build a blackjack card game which allows for a game of 3 players to play against a computer dealer.

The board should be made up of a minimum of the computers cards, and up to 3 players hands. The cards may always be shown. Newly drawn cards should be added to the active players hand.

1-3 players may play the game against the computer simultaneously each taking their turns respectively

At least 1 player must enter the game to start a game and a username will be required upon entry.

Each new player will start with 100 points. If an existing user signs in, they will retain their existing points. This should persist even if the app is shut down and rerun.

Each game entered will wager 10 points, if the player wins, they win their original 10 points plus an additional 10 points.

If they lose, they will lose the 10 points. When a player is out of points, they can no longer play the game. If they try to play the game again it will say no points available.

When it is each player’s turn, the active player will be able to press either “hit” or “hold”. “Hit” will draw another card to the pile, and “hold” will continue onto the next players turn.

Pressing the “hit” button will add a card to the active players hand.

If the player goes over 21, they will no longer be able to click the “hit” button and it will be the next players turn.   
  
If the player presses the “hold” button, the players turn will end, and the next player will be able to press the “hit” or “hold” button.

The game should allow a user to play again or exit their game after each hand. If the user choses to play again, they should not be required to log back in.

If a user exits the game, a new user can sign in to take their seat.]

            Each game is one round

            Players can enter and exit when a new round starts

            Note: The **dealer** must draw another card if the value of their hand is lower than 17.

# Team Members

Brandon Escota, GUI/Front End design.

Max Nelson, Game Controller/Driver.

Wilson Pena, Users/Player data.

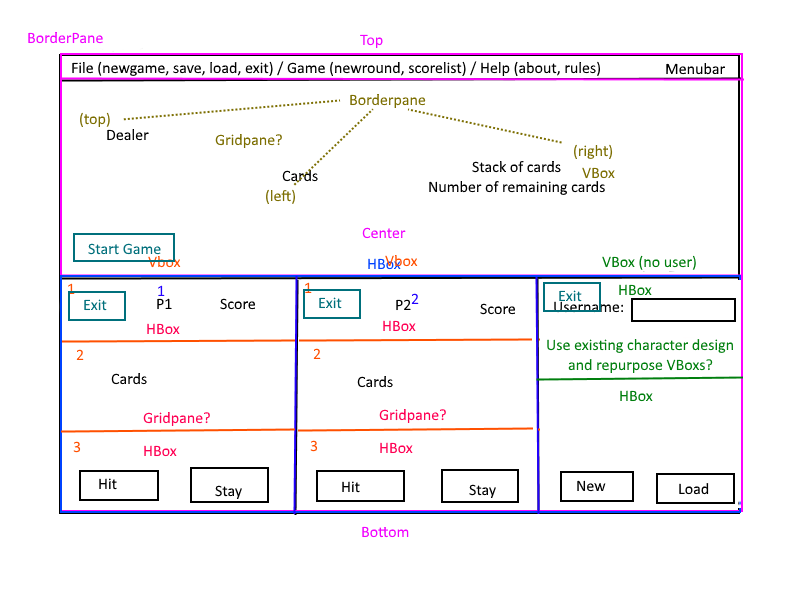
# Deliverables

This project outline, sprint log, all Java source code and Users Guide/Documentation.

# Features

1. Use a single card deck to allow from 1 to 3 players to play black jack against the computer.
2. Each player will begin with 100 points. Each player’s points will be stored each time they play.
3. Every time they login they will continue with the points they were left with during their last game.
4. Each Game will require a 10 points wager, if the player wins they will keep their wager and get 10 for their win. A player will not be allowed to play once they have lost all their points.
5. All players hands will be displayed in the GUI.
6. Each player will sign in with a simple username.

# Application Wireframes



|  |
| --- |
|  |
| * Describe the functions that the wireframe will contain  1. Dealer pane where the user will start a game, when at least one player is logged in. The dealer will hit if it has less than 17 points. The dealers turns comes after every logged in player has had their turn. 2. All three players have an identical pane, with and exit, hit and stay buttons. |
|  |

# Results

There were 7 total sprint runs.

# of hours / points completed in total:

Total hours 12 hours.

# of hours / points completed each sprint

Each sprint we completed 2 hours of work.

1 – Sprint 1 – Day 0

During this sprint, we discussed the game design and work required. The assigning of each team members responsibility and part to be completed was done. We created a Trello account which is a collaboration tool that helps to organize a project into boards and cards. Trello was used for us to see who was working on what and to scope all the work required. We also created a GitHub repository for all out code and documents.

2 – Sprint 2 – Day 1

Created all cards for each team member’s part. Assigned preliminary due dates for all cards.

3 – Sprint 3 – Day 3

Continue to work on Individual tasks/parts. Discussed questions regarding the merging of all the individual parts and updated all team members of current status of project parts. QA regarding all required methods input and output. QA on how the final Blackjack should look, operate and all the individual parts should merge together.

4 – Sprint 4 – Day 4

Continue to work on Individual tasks/parts. Discussed questions regarding the merging of all the individual parts and updated all team members of current status of project parts. QA regarding all required methods input and output. QA on how the final Blackjack should look, operate and all the individual parts should merge together.

5 – Sprint 5 – Day 5

Continue to work on Individual tasks/parts. Fixed issues with displaying rendering main UI and the displaying dealer and player cards. Fixed pane issues with not displaying playing cards properly. Discussed questions regarding the merging of all the individual parts and updated all team members of current status of project parts. QA regarding all required methods input and output. QA on how the final Blackjack should look, operate and all the individual parts should merge together. Integrated player, UI and game code with just display cards for all players including the dealer.

6 – Sprint 6 – Day 6

Continue to work on Individual tasks/parts. Fixed issues with displaying rendering main UI and the displaying dealer and player cards. Fixed pane issues with not displaying playing cards properly. Discussed questions regarding the merging of all the individual parts and updated all team members of current status of project parts. QA regarding all required methods input and output. QA on how the final Blackjack should look, operate and all the individual parts should merge together. Integrated player, UI and game code with just display cards for all players including the dealer.

7 – Sprint 7 – Day 7

Final bug fixes and testing, completed Blackjack project.

Team Velocity: 16 points per each sprint. We had six sprints.

1 – Decided to use Trello and GitHub for organizing and code repository. 10 points.

2 – Created all cards for each team member’s part. Assigned preliminary due dates for all cards. 10 points

3 – Continue to work on Individual tasks/parts. Discussed questions regarding the merging of all the individual parts and updated all team members of current status of project parts. 15 points

4 – Continue to work on Individual tasks/parts. Discussed questions regarding the merging of all the individual parts and updated all team members of current status of project parts. 15 points

5 – Continue to work on Individual tasks/parts. Fixed issues with displaying rendering main UI and the displaying dealer and player cards. Fixed pane issues with not displaying playing cards properly. 15 points.

6 – Merged all code and tested, preliminary attempts on playing a full game. 20 points.

7 – Tested final code and worked on fixing bugs. 15 points.

Variance Per Sprint (14 hours estimate ( total hours estimated )/ 16 total actual hours )

1 – 2hrs estimated / 2hrs Actual

2 – 2hrs estimated / 2hrs Actual

3 –2hrs estimated / 2hrs Actual

4 – 2hrs estimated / 2hrs Actual

5 – 2hrs estimated / 2hrs Actual

6 – 2hrs estimated / 3hrs Actual

7 – 2hrs estimated / 3hrs Actual

|  |  |  |
| --- | --- | --- |
| Brandon | Max | Wilson |
| Figure out this notepad stuff | Figure out this notepad stuff | Figure out this notepad stuff |
| PlayerPane with <User> myUser, <int> points, <Hand> handOfCards, <boolean> isOnline; has methods Exit, Login, UpdatePoints?, Hit, Stay, ShowHand | 52 cards, consider a different deck with better dimensions. Has integer value. Ace = 11. Also has back of card image, and boolean variable deciding whether card is face up/down. | User object with name and points. |
| LoginPane that takes in a <String> name, and depending on what Button is pressed, will either create a new player or load a player. If save file is internal, have handling for if a duplicate player exists in the save file. | Deck of cards class, contains a deck of cards ArrayList of type card. Methods shuffle, draw, and fill deck (resets deck to default state) | Save data with name and points, extends however long we want it to. Can generate new file or load existing file. Internal is ideal. Basically stores the User object. |
| DealerPane with a visual stack of cards; methods ShowHand, StartRound, Hit, ShowScore | Hand class, contains an ArrayList of cards that a user is holding. Has the methods draw, getValue, and compareTo(Hand). | UserList, list of users. Can find users by name and return them to the player pane, creates a new user with 100 points if no user by that name exists, then returns that. |
| Card rendering / middle-node of PlayerPane | Card backs for the deck of cards  Card back is .jpg, but card images must be .png | LoginPane that takes in a <String> name, and depending on what Button is pressed, will either create a new player or load a player. If save file is internal, have handling for if a duplicate player exists in the save file. |
| Button methods for PlayerPane, LoginPane, and DealerPane. | Cleanup card folder, remove copies, delete current card image folder from github, add a link to drive folder of cards and rename cards | Create and update project-outline. |
| Pane Swapping | Card rendering / middle-node of PlayerPane | Create and maintain |
| Artistic bits in the middle of LoginPane to fill in empty space? | Deck Class: Variable 'folder' path should be consistent (like ./Cards/), not something to be passed in. Be sure to comment where the folder should be located on local workstations so it can be loaded correctly. |  |
|  |  |  |