Project Title: Joker

Team name: Jokers

Members:

Chen Huang: chen_huang@ucsb.edu ← The alpha
Zeyu(Harry) He: zeyuhe@ucsb.edu
Zhijun Yan: zhijunyan@ucsb.edu
Nan Wang: nwang01@ucsb.edu
Shida Sheng: shidasheng@ucsb.edu

What the project is about?

What problem the project is solving?

Everyone needs entertainment and gambling is a classic way to reduce stress, but Las Vegas is way too far. Why don’t we gambling on mobile devices. So, we decided to make a single casino game that can be played on PC. Since we don’t involved in real world money, we create an easier, faster, safer way to gambling at home, at work, at toilet, etc..

Why the problem is important?

Since nowadays the mental health is of crucial importance and being bored and stressed out can lead to unwanted effects, this kind of online games can make people cheerful and in good mental health condition.

The outcome of the project

A recently very popular casino game, BlackJack. It is a 2D game that can be played on mobile devices with vivid vision and sound. Every player is automatically assigned an account with 5000 dollars and can play the game. The money he or she gains or loses can automatically be calculated by the system and appears on his or
her account. In this way, players can enjoy this game with fictitious currency at home at any time.

Milestone
stage 1: implement Black Jack game → python (2nd, 3rd week)
  a. the logic of the game
  b. bet and win money calculator
  c. player's account building
stage 2: visualize the game → pygame (4th, 5th week)
  a. build the user interface that can get input from the user by graph instead of command line
  b. visualize different kinds of cards
  c. visualize what cards are at hand and the fictitious money
  d. show the player profile (money remains, profile picture)
stage 3: improve the visualization(adding sound and motion) → pygame(6th week)
  a. add the motion of the card (drawing the cards and open the cards)
  b. add the sound of the game (background sound, hit sound, open a card sound, winning sound, losing sound)
possible stage 4: try to achieve multi-players and networking
possible stage 5: try to use the classes for Black Jack to implement other card games.

Process
stage 1: Implement Black Jack game → python(3rd, 4th week)
  a. The whole logic of the game is built
b. The game can run successfully

stage 2: working on new features according to user stories (5th, 6th week)
   a. Test for each new feature
   b. Working on card visualization
   c. Revise code and give more useful information while player playing games

stage 3: Revise the code and put our code into pygame frame, visualize the whole program.
Use case (Revised)

1. As a player, I want to login in this game so I can start to play the game on my mobile devices.
2. As a player, I already logged in to my account, I go to the main menu directly where I can choose to check my balance to see how much money I remain or start a new game.
3. As a player, as I start a new game, I can decide how much money to put in the next game when I already know my balance so that the next game can get started.
   a. Task 1: Create a basic program that accepts user input. Prints start the game when user inputs ‘start’.
      i. Estimated time: 1 day
      ii. Who is assigned to this: alex
      iii. Go right test: the game will ask: how much you want to bet
           the player type in a number from 1 to the money he has
           the game will say: ok you bet ____ and game started
iv. Exceptional test: the player type a number out of range or other words the game will say: please type a number from 1 to ___

4. As a player, when the game starts, I want to see what card I have and what card the dealer have so that I can think about if I need to draw another card.
   i. Estimated time : 1day
   ii. Who is assigned to this:
   iii. Go right test:
       the dealer draws a card [***,***]
       you draw a card[diamond, 5]
       the dealer draws a card [diamond, 8]
       you draw a card [heart, 6]
       now you have point 14 in total
   iv. Exceptional test:

5. As a player, after the game starts, I want to decide whether or not to draw another card so that I can let the sum of my cards close to 21
   i. Estimated time: 1 day
   ii. Who is assigned to this: Harry
   iii. Go right test: the terminal will output an prompt: “ Do you want to hit or stand?” then the user will input: “hit” or “stand”. If “hit” is the input then we print the new random card on the terminal: “The new card you draw is:__”
   iv. Exceptional test: If the user input something else other than the two input, the terminal will print out “please specify do you want to hit or stand”

6. As a player, I want to see what new cards I draw and what is the total number I got so that I can decide if I need to draw another card.
   i. Estimated time: one day
   ii. Who is assigned to this:
   iii. Go right test: After you type “Hit”(or choose to draw a new card), the game will show the information of new card. It will automatically show the total points of cards you have. If points are over 21, game over. And go to step 8. If points are close to 21, the game tells you whether you want to draw another card or not (go back to step 6)
   iv. Exceptional test:

7. As a player, after a round of a game, I want to see if I win or lose and how much money I win so that I know how much money I have right now.
   i. Estimated time : 1day
   ii. Who is assigned to this:Kenneth
   iii. Go right test: Dealer show card, if total number is smaller than17, dealer automatically draw another card. And show all the cards on table(dealer and player); show result and then the game shows win or lose and followed by the statement that shows the money you win from the game and also the total money you have currently
   iv. Exceptional test: : If total is 0, kick out or “Do you want more token(chip)?”
   v. As a player, I can input in terminal “RESTART” and return to step 4 to decide how much money to bet.
   vi. Estimated time :1 day
   vii. Who is assigned to this:
   viii. Go right test: a player can whenever input “RESTART” on the terminal, then the current game will terminate and the computer will ask “how much you want to bet”, which means that the game progress restart from the beginning(back to step 4)
   ix. Exceptional test: the player type things other than “RESTART”, the game will not restart. If the word the user typed can engine other command, then other command will
8. As a player, I can input in terminal “RESTART” and return to step 4 to decide how much money to bet.
   i. Estimated time : 1 day
   ii. Who is assigned to this: Nan
   iii. Go right test: a player can whenever input “RESTART” on the terminal, then the current game will terminate and the computer will ask “how much you want to bet”, which means that the game progress restart from the beginning(back to step 4)
   iv. Exceptional test: the player type things other than “RESTART”, the game will not restart.
9. As a player, I want to stop the game anytime I want because sometimes I may have other things that prior to the game to do.
10. As a player, I want to see what card I draw because this can offer me a real game feeling.
11. As a player, I want all my game data in one account can be reserved online because in this way anytime when I log in this account on any devices, I can continue my gaming process.
12. As a player, I don’t want to play this game alone, so I would like to add some other AI player to play the game with me.

Retrospective Info
1. terminal based game
2. login and database
3. visualized basic game from pillow to pygame
4. our final version

In Sprint 1, we focused on the terminal based game, we completed a well-organized game logic code. We also make our code to calculate the total points in player's and dealer's hands. Then it outputs win or lose. Basically in sprint 1 we completed our game blueprint without any other features. Then in Sprint 2, we focused on game data and other features like login. Based on these things we also had new features like ranking according to the stored data. Also, we started to learn about game visualize like pillow. In Sprint 3, we realized pillow cannot bring us what we want, so we shifted our focus on pygame. And in Sprint 3, we tried to visualize our game with pygame and we made it. After these sprints this quarter, we successfully visualize our game and make it functional.

Challenges Faced / Overcome
The major challenge is that we chose to reinvent the wheel and not to use various fancy API or game engine. The reason is that by doing so, we might get a deeper understanding of game developing process and also in this way it's more convenient for us to develop our customized feature outside the rigid constrain of different game engines. The second largest challenge is that we spent a lot of efforts to get used to Agile Software Development. At first, we wanted to have something workable as soon as possible, thus we spent a large amount of efforts building the terminal version. We have to first set aside the graphics part and make every feature workable on the terminal version. However, when later we tried to build the graphical interface, we found that most of the game functions are not usable for the graphical interface version and had to rewrite the whole structure of the game. This challenge is due to that we had a hard time to get used to the agile development. We were overly eager to have the code workable at every single stage and used many global variables instead of member variables and member functions, so didn’t plan very far for the graphics parts and wasted a lot of efforts for the terminal version. The minor challenge is the learning curve of library pillow and pygame. We thought pillow is more like modifying the pictures thus we turned into pygame.

Missing / Remaining Features or Functionality
Features we have:
1. User's information, including balance, account name, password, win time, lose time
2. Users can choose bet amount before each round of games
3. Show the total points in game
4. Visualization of cards, make buttons to visualize user's actions
5. Player ranking according to their balance and win/lose times

Feature we're working on in the future:
1. 3-demensional background animation
2. Functional database
3. Functional AI players with different game logic

About Test

graphtest.py
bjw.py
importImage.py
player_hand_status_test.py
playertest.py
testcard.py
total_test.py
.travis.yml

During the whole development of our game, we tried many different ways to test our code to make sure it works. The names of files shown above are the test files we used during springs. Basically they are used for checking the game logic works well or not. Some of these files have edge cases for calculating card points, and some of these files are used to test our code can visualize the right card every time the code calls a card. And we did use travis CI several times, but we didn't have much progress on our code with travis CI so we shift to manually tests. And manually test is the main way how to test code.

In our project, the game logic, cards visualization, total card points have all been tested. Many of them are tested with test files but we also found that manually test also works. In our project, in fact, we think manually test is more efficient because we have our daily meeting and code together, we can check the error messages every time we test and put our focus on the lines where error messages showed.
High-level Architecture Diagram.

USER

push request

terminal

I/O

Visualization

Invoke

Call

Game

Search Information

Continue

database

Fetch Information

Player

Team: Joker
Member: Alex Huang
Harry He
Kenneth Yau
Nan Wang
Skida Shao
button

player

playerHit() → hit()

reset() → reset()

a Carol object is created

the Carol object is destroyed

button and player object already exist
Necessary link:
github: https://github.com/Alexxx411/cs48_jocker
slack: https://joker-nfm5441.slack.com/
trello: https://trello.com/b/ie5HjcwH/jocker
presentation: https://docs.google.com/presentation/d/1q3CKijEcA_MKMgl1gGCZXLWmOBVR- EVzUppNa-0G_c/edit#slide=id.p
Link to the recording of demo in class
https://github.com/Alexxx411/cs48_jocker/tree/master/demo
Appendix 1

Commit History
Commits on Jun 12, 2019

Add files via upload

  shidasheng committed 6 minutes ago

Delete user.exe

  shidasheng committed 6 minutes ago
Delete user.exe

  shidasheng committed 6 minutes ago
Delete readme

  shidasheng committed 8 minutes ago
Add files via upload

  shidasheng committed 10 minutes ago
Add files via upload

  shidasheng committed 12 minutes ago
Create 1

  shidasheng committed 12 minutes ago
user part for single version

  zeyuhe committed 13 minutes ago
user part for muti version

  zeyuhe committed 14 minutes ago
Create 1

  shidasheng committed 14 minutes ago
Delete 1

  shidasheng committed 16 minutes ago
Add files via upload

  Alexxx411 committed 1 hour ago
Add files via upload

  zhijunyan committed 1 hour ago
Add files via upload

  zhijunyan committed 1 hour ago
Add files via upload

  nwang01 committed 1 hour ago
Add files via upload
Commits on May 28, 2019
fix the bet feature minor bugs

Commits on May 24, 2019
can use the feature hit

Commits on May 23, 2019
Add files via upload

Commits on May 22, 2019
Delete cards.zip
the path of the game

can create account
zhijunyan committed on 13 May
Delete d8.jpg

zhijunyan committed on 13 May
Delete s10.jpg

nwang01 committed on 13 May
Delete cj.jpg

zhijunyan committed on 13 May
Delete s2.jpg

nwang01 committed on 13 May
Delete s4.jpg

nwang01 committed on 13 May
Delete s5.jpg

nwang01 committed on 13 May
Delete cq.jpg

zhijunyan committed on 13 May
Delete s6.jpg

nwang01 committed on 13 May
Delete d10.jpg

zhijunyan committed on 13 May
Delete s7.jpg

nwang01 committed on 13 May
Delete ca.jpg

zhijunyan committed on 13 May
Delete s8.jpg

nwang01 committed on 13 May
Delete c9.jpg

zhijunyan committed on 13 May
Delete s9.jpg

nwang01 committed on 13 May
Delete h10.jpg

zhijunyan committed on 13 May
Delete sj.jpg

nwang01 committed on 13 May
Delete sk.jpg

nwang01 committed on 13 May
Delete c8.jpg
Commits on May 13, 2019

Delete sa.jpg

Delete sq.jpg

Delete d7.jpg

Delete c7.jpg

Delete d3.jpg

Delete c6.jpg

Delete s3.jpg

Delete c5.jpg

Delete c4.jpg

Delete h3.jpg

Delete c3.jpg

Delete c2.jpg

Delete c10.jpg

Commits on May 10, 2019

Sprint 2 Burndown Chart

Delete c10.jpg

Commits on May 9, 2019

Add files via upload

Update .travis.yml
Commits on May 8, 2019
Update README.md

Commits on May 3, 2019
Add files via upload
Add files via upload
Add files via upload
Update README.md

Special cases test for player's hand

Commits on May 2, 2019
Add files via upload
Delete untitled.cpp
Add files via upload
Add files via upload
Update README.md
Merge pull request #2 from Alexxx411/harry
this version will give interface
Update .travis.yml
Create .travis.yml

Merge pull request #1 from Alexxx411/master
Update .travis.yml

Appendix 2

Technology we used:
- Pillow
- Pygame
- C++
- Python
- Travis CI