Analysis

Problem:

Young people looking to play poker often have a lot of roadblocks and hurdles getting in their way. Whether this is a lack of interest from their friends, or more recently lockdown, meaning they can’t physically sit down and play with people or online solutions requiring they are above a certain age or having predatory monetisation schemes designed to take advantage of young people and designed to addict people using subconscious techniques. This leaves a giant gap in the market for accessible and non-predatory solutions that allow young people to practice and play poker in a safe environment.

How I can help by using computing:

The most obvious solution to not being able to sit down and do something is to use the internet to find a way. Without a custom solution such as mine it is impossible to play poker online as you need to play using one deck of cards and passing chips between each other. However, by using a computer and the internet you can easily have people sending data between each other with negligible delay, this is almost impossible with any other solution. Therefor we need to use computers and the internet so that data can be sent. Furthermore, due to the wide availability of the internet and the ability to store data on a central server people can log in from anywhere and pick up where they left off and see all of their data at any time without needing to bring any physical media with them. This is another task that is impossible for human’s as they would need to physically bring some method of recording data and the physical data with them to wherever they are playing from.

In addition to this the storing and tracking of data is another difficult task for people to do manually as they have to remember how and what they have done and manually plot graphs and make calculations. This wastes time that people want to be playing the game. On the other hand, this task is extremely well suited to a computer as they are designed to be able to handle data as they can easily take inputs then store, process data and display data automatically all the while unimportant information like how these numbers are generated is abstracted away to create a better user experience and save time.

Furthermore, by creating my own solution from the ground up using computational methods I can control the monetisation and the safety of the environment to ensure that no one is being taken advantage of and the game is more appropriate for all ages. This can be done by limiting the interactions between players to purely gameplay based or, if a game chat is included, to ensure it censors inappropriate messages. This isn’t possible if you are playing with strangers in real life as you can’t predict or control their behaviour or how they communicate however computers can easily parse messages and check for and violations before they are transmitted.

Stakeholders and the target demographic:

The product is targeted at anyone who wants to play poker online in a safe environment, but primarily under 18s who are unable to access many other due to restrictions in place or just because their parents/carers are worried about the dodge monetisation methods of some commercially available solutions. Despite this being the target demographic it will be accessible and appropriate for anyone from ages 12 and up to use to practice their skills without real currency on the line. It is aimed at new and experienced players alike as it will have information on how to play and what the rules are accessible if someone wants that information. It will also be aimed at both casual and competitive players, the open table format opposed to a tournament format targets it at casual players however the extensive statistic and data analysis features means more serious players still have a reason to use the platform. Because of this I will be interviewing and testing the product on a range of people under the age 18 with varying levels of experience with poker.

Interviews:

## Person 1 – Felix Judd, 17 years old:

Q: How would you describe your level of experience/interest in poker

A: I wouldn’t say I’m experienced; I played casually with my family a handful of times a few years ago. I am definitely interested in learning more about how to play though before I even consider gambling actual money

Q: Have you ever used an online poker platform before? If so, what were your thoughts

A: No, never used anything like that before

Q: Why not?

A: I didn’t realise games that weren’t for real money existed

Q: what features would you look for in a poker game?

A: As someone who doesn’t know much about poker a built-in hand ranking and maybe a tutorial or some hints or something just so I don’t feel totally out of depth, also having it be simple so I can understand it quickly

Q: How interested are you in the analytical side of the game?

A: Not very interested to be honest, I guess its interesting to look at but I’m not really sure what I’d do with it

## Person 2 – Louis Gosling, 17 years old:

Q: How would you describe your level of experience/interest in poker

A: I don’t play a huge amount but a few times a year my family plays a fairly serious game and I have played a few free online games and watched a few videos and all that. So not super interested but definitely not a beginner and want to play more and get good again before uni and stuff like that

Q: what features did you like in the online version you played?

A: I liked that it had an overriding currency and it was individual for each game, it gave it a feel of continuity and meant I could easily see my successes as it is reflected in my bankroll graph

Q: So building on that, how interested are you in these statistics and analysis features

A: Very, its something you obviously don’t get in real life but its so nice to have. Especially when using it as a practice tool the ability to see the trends and mistakes I’m making is really useful

Q: What would you change about the version you used?

A: The ui kind of sucked, it felt very overwhelming to use and I basically only pressed 25% of the buttons so something more simplistic without all the extra unnecessary buttons and stuff would be great

## Person 3 – Katie Curtis 19 years old:

Q: How would you describe your level of experience/interest in poker

A: I have played really casually with my friends a few times, its fun but I’m very causally interested just as something to do

Q: Have you ever used an online poker platform before? If so, what were your thoughts

A: No never

Q: What features would you look for in an online poker game

A: the ability to play with my friends is all I’m really looking for, and something to tell me how good my cards are, normally I have to google that

Q: How interested are you in the analytical side of the game?

A: not at all, I’m just wanting to relax and have some fun with my friends because we aren’t really video game players and this would give us an opportunity to play together while we’re away from uni because of holidays or lockdowns or whatever

Conclusions from interviews:

There is a large variety of skill and interest levels around the game of poker and the ability to play it online with different people wanting different things. Both of the less experienced players wanted a ranking of the hands to be easily available and built in so this is definitely something I will aim for. The analysis section didn’t seem to be as a big a deal as I had hoped however it seems essential for getting more serious competitive players to take interest in a more basic solution. Maybe some idea of what should be expected/ is a good sign in the analytics would help people like Felix take more interest in these statistics and develop their game since the only reason he gave for not caring was a lack of understanding. The desire to play with friends means some way of identifying who is hosting a server is important overwise it would be hard to coordinate a game amongst friends, maybe the ability to have a private lobby with a password would help keep the game among friends. A simple ui also seems important to people as both Louis and Felix mentioned it, and one can only assume that a more casual player like Katie would also appreciate a simple and easy to use interface, especially around the server browser. This mostly reflects what I expected to be concluded from these interviews, but I had my priorities out of order initially, over valuing sections like statistics.

## Important points

* Clean ui
* Clean and easy to use assists for new players
* Statistics with explanations of what they mean for the player
* Named lobbies
* Low barrier of entry so new players don’t get overwhelmed

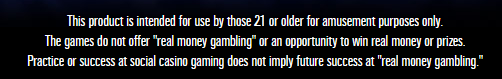
Research into pre-existing solutions:

## The official world series of poker game:

Cons:

* Game is “aimed at people over 21”
* A very busy and obnoxious user interface
* Focussed around the in-game store where you have ability to pay real money for their virtual currency.
* In game store offers an advantage to some players who have spent money.

Pros:

* In-depth tutorials and rule information
* Very in depth, if hard to find, statistics such as stats on when the player commonly folds.

Overall:

Not a bad solution with high production value and the ability to log on with several options as seen above. Overall though they clearly state they have different stakeholders for the product and are very heavily focused on monetisation. This is not something I am interested in as it will put off younger users and/or their parents. I will certainly use the statistics from this solution as an inspiration for my own statistics model and what to include. I will also aim to make a more simplistic user interface as I found this one hard to navigate and very claustrophobic.

## Pokerist:

Cons:

* As seen here the user interface puts a large focus on promoting in-game purchases
* Limited statistics, only game played and biggest win
* Use of overly sexualised characters showing a clear target demographic
* Due to being web based a long loading process before getting to the menu
* Hard to select the right number of chips
* Strategies designed to addict such as daily bonuses to encourage people playing every day
* The fold call and check button are relatively far from each other causing a sub optimal ux.

Pros:

* Once you close the ads it reveals a very clear and useful server list displaying relevant information such as players blinds and buy ins
* Intuitive hints that show when you have a hand which can be disabled in settings
* Use of accounts so information can be accessed from anywhere
* An in-game history so you can see the result of the previous hand
* Can preselect your next action in advance so it will complete automatically

Overall:

A better solution than the wsop but not without its flaws. The user interface was, other than the large advertisements covering up useful features, very clear and much less claustrophobic than the wsop game. This is likely due to its use of clear tiles and the ability to see the plain background behind it. The game also has a lot of social features which I found unnecessary as it detracted from the game itself.

## 247 free poker

Cons:

* No online component, purely against ai
* Almost no statistics at all (only largest win)
* No settings/preferences
* All data stored in cookies, can’t access from another device
* Doesn’t say if you have had a hand, no assists
* Could be clearer about rules
* Doesn’t say how much you are calling

Pros:

* No monetisation or adverts at all
* Complex/competent ai with a consistent play style you can learn so the game retains depth despite lack of real people
* Very simple ui no visual clutter at all
* Appears to have similar stakeholders to my game in objectives just without the online component
* Clearly made just for people to enjoy/practice no ulterior motives

Overall:

This is probably the closest to what I want to create, very simple no nonsense design. Clearly designed for similar stakeholders to mine. Obviously, the lack of an online component and statistics hurts it, but I believe in terms of the basic user experience this is a realistic and good target for my product

Observations of research

* Free poker games that aren’t focussed around real world money and gambling appear to web based in the browser or mobile(very few downloadable executables), while my lack of CSS/js knowledge will prohibit me from breaking into the in browser market, a mobile port of my game may not be too hard and is certainly something to consider
* These games are very focussed around getting money, whether by promoting an in-game store or external betting platforms
* There is much differentiation between the casual open tables and the organised tournament games
* The in-game user interfaces are all designed to look like a poker table
* Simplicity is key, most of the depth in poker comes from the players not the platform that people are playing on

Key features observed

* Assists to show hands
* Statistics
* Online play
* A clear and simple ui
* A good server browser with options/information about the game your joining
* A login system so information can be accessed from anywhere

Proposal

Due to the fact that none of these solutions (or any other that I found but did not write up) met all the requirements I had, and my stakeholders outlined. I propose creating an online (but not web based) poker game based around an artificial currency with no links to real world money at all. The solution will have a server browser where people can connect to a game of their choosing so they can play how they choose. The game will also have extensive statistics and analysis features, such as past games overall bankroll and other useful and relevant statistics, so that people can look at their game and learn from the experience, this is important as a large portion of stakeholders are people wanting to learn, improve and understand more about the game and what they are doing right and wrong. Another important aspect I will include is a secure login system so people can assess the game from any device. Having an online system means password security is more important than ever and not having proper password security could put me in breach of the data protection act. I will also try to include a decent tutorial or, at the very least, assists that show the information about your hand and what you have. For example, highlighting and pointing out pairs and straights etc. I also have some aspirational targets for if I have the time and ability to add such as organised more competitive tournament games that people can put their skills to the test in as well as interactive replays.

Software limitations

Due to the cost of actual server space I will have to find a more cost-effective solution to hosting my logon/game server. This will likely be in the form of a raspberry pi which will limit my space and memory meaning I won’t be able to have the large number of users that would be expected from a larger commercial competitor. I will also endeavour not to have to use this server for routing all game information which requires either getting users to port forward, which is a very ugly solution, or figuring out UDP hole punching. Other software limitations are the very server/host side nature of the processing that is necessary to coordinate all aspects of the game, therefor the hosts pc will be slower and have to constantly be sending and receiving packets, which is not a fast process. It would be nice to integrate this into a webapp however I do not have the knowledge or ability to do this with the time limitations of this project as I would have to learn it from scratch an integrate some form of cross platform support. On the other hand, I may be able to have a mobile version as there are many tools for packaging pyqt5 applications for android and all I’d have to change is any operating system file directories used. However, even if I did port it for mobile, I still would not put it on any app stores as this requires licensing and is especially difficult around gambling games. This makes it unfeasible for the time and economic constraints of the project.

Hardware limitations

The game should not have any serious performance issues running on most devices due to the simple nature of the game and design, this could still be optimised with threading if lower powered devices appear to struggle. Furthermore almost all actual processing being done by the host pc as long as they have a strong pc almost anyone should be able to play. The reliance on networking means that people who don’t have an internet connection cant access anything, not even their statistics as they would need to log on, which uses the internet, and access the data itself, which is stored on a server and so needs the internet. Due to using low level sockets I will also need to send raw data across the network rather than images or pre calculated graphs. This will take up more memory on the user’s system when looking at statistics but should not affect the actual gameplay itself. Any dedicated server hosting options cost over £40 a month which is above budget for this project, therefor for testing I will use my desktop pc as a server and will migrate it to a lower running cost low power draw raspberry pi or similar solution.

Software choices/evaluations

For this task I chose to use python for several reasons. Python is one of the most versatile languages currently on the market allowing you to combine object oriented programming and procedural programming in one file, this allows me writhe my code with very few external restrictions allowing me to rapidly prototype and iterate upon solutions without having to commit a lot of time to the codes format and style early on. Python also built around the idea of a “program” being built around several modules, this is useful for me as at different points in the runtime the program has to be doing very different things and separating these out into sperate files will make the development much easier and improve code maintainability. Python does have the disadvantage of being significantly slower than its competitors such as c/c++. This is because of two main reasons, one of which is pythons interpreted nature as opposed to c/c++ which are both compiled languages. This means that, while you can test prototypes much faster in python as interpreting code is faster than actually compiling code, once compiled it is still actively translating from the high level python into machine code whereas compiled languages have a fully written executable of machine code ready to go which is obviously much faster. When packaging pyQT code it is actually converted to c++ so hopefully I will gain some performance back, but it will never be as efficient. Python is also a very high-level language meaning you have less control over memory and pointers, this is great for security and robustness but can lead to slower less efficient programs than its lower level counterparts. Despite this its still the best option for me as, not only do I not have time to learn a new language, python is perfect for the iterative agile development style I am aiming for due to the speed you can get a prototype working. Not only this but as it is interpreted cross platform support with mobile or other operating systems is far easier than it would be with a compiled program which leaves this avenue open if I choose to pursue it.

Python also has a wide range of third-party libraries I will be making use of. Such as the socket module which provides an easy to use low level networking interface built off of WINSOCK. This means the networking features are widely supported and used even outside of python applications allowing me to find information and support from a wider variety of sources. I will also be using the pyqt5 library to create my user interface, this is great as it is built off of the widely used qt platform so includes support for the qt designer, a great tool that allows you to create and edit the user interface while seeing the result in real time unlike something like tkinter or CSS where changes are made in code and it has to be run to see what it looks like. pyQT also has many built in “widgets”, these make getting a functional user interface that conforms to the systems aesthetic easier than ever, as well as having built in events and methods to make displaying and receiving data a very simple process. I will also be using SQLite for my database manager. I made this decision as I already had modules and functions written in SQLite and sending the results of queries over a network using serialisation from the pickle module was an easy and fast solution in testing. A system like mySQL could have been better as I am using for server like processes, however the learning curve and additional code made it hard to justify when I already had a system in place and was on a tight time budget for this project. I will also be using python’s cryptography module as the popular “pycrypto” module is no longer being updated and so is not secure enough for my use case.

By making almost everything processed server side it increases the security as any inputs or changes to the database can be extensively checked and verified before being run whereas storing information client side can cause risks such as someone accessing the raw database without logging in and viewing/changing personal data. There are still potential security threats though, by having an open port and public ip it is important to set up validation of all incoming packets on the server and making sure no message is ran into sql code directly as this opens you up to an sql injection attack. For further security, a ddos protection service such as “cloudflare” could be used. However, with the scale of this project I decided this was unnecessary but if I was scaling it up it is something, I would have to consider.

Stakeholder response to proposal

I gave my stakeholders a chance to read the proposal paragraph of this document and overall they seemed happy with it and felt their concerns had been met and that if a solution like that was on the market for free they would use it. They did comment on the fact that I would be making a downloadable and not web based version, they said that this was a downside as downloading unknown files feels riskier than using a website but that if they were using regularly the convenience of a web based service hits diminishing returns. The more casual players (Felix Judd and Katie Curtis) said a mobile app would be great whereas Louis Gosling (the more competitive/serious player) said he didn’t see the appeal of a mobile game as if he’s playing poker he wants to be focussed so would be sedentary anyway making the benefits of a mobile device lesser.

I don’t think these responses will change my proposal at this point but gave me positive affirmations I’m on the right track.

Success criteria

## Out of game:

|  |  |
| --- | --- |
| Objective | Evidence |
| User can Join a game from list | The server list displays all available games, the user can make a selection and then be entered into the game |
| User can create an account | User can press “create account” button and then their details and password will be hashed salted then sent over the network and stored correctly |
| User can login to an existing account | User can use details associated with an already existing account which will be hashed and salted before being sent to the server and compared, if the details are correct it should allow you to enter the application |
| User can logout of their account | Once logged out any data that had been stored temporarily is removed, sends you back to the login screen |
| User can view statistics of their games | Users can select an option from the main menu from which they can see key percentages |
| User can view their overall money | Overall money is displayed on the menu |
| Once bankrupt money resets | Once the player runs out of overall money (not individual game) their currency resets and this is recorded in the stats |
| User can host a server | User can select an option to host a server which is added to the server list and which people can join |
| An intuitive GUI (graphical user interface) | No console/command line is needed to access any feature. A user with any level of computer/poker experience can easily navigate the menus and do what they want to do |
| All statistics are stored on server’s database | Any statistics are stored remotely and associated with the account rather than stored locally. |
| Data should be protected | Any personal data stored such as names, dates of birth, emails etc. Are stored and transmitted safely and securely in accordance with the data protection act |
| User can fully delete account | The user has the ability to fully remove their account from all servers and databases if they choose to and enter their password |

## In game:

|  |  |
| --- | --- |
| Objective |  |
| Players can see each other’s actions | Once a player makes an action this is reflected on everyone’s screen not just theirs |
| Players can place bets | * Players can choose to bet currency on a round which is then drawn from their stack, * If they bet more than they have they are put all in, * they can only bet money they have. * Money one is added to their stack if they win * Cannot bet less than the big blind |
| Players can fold hands | A player can fold a hand and is then no longer prompted for more actions and receives no money |
| Betting rounds continue until everyone is folded, all in or even | The betting doesn’t stop until everyone has either:   * matched the largest bet * only one player remains in which case they win * players haven’t matched but are all in |
| Pots are allocated correctly | At the end of a hand money is allocated correctly in accordance with any split pots that may or may not have formed |
| Winners are calculated correctly | Once a round is played the winning hand is calculated and displayed correctly and all hands are ranked so that split pots can be assigned in order |
| Players should be able to leave whenever they choose | A button which when pressed exits a player to the main menu and updates their statistics |
| Large and small blinds are taken correctly | The blinds should   * be taken automatically regardless of player action * Increment every time the dealers button does a full rotation * Be rotated by one place every round * Notify players when they are the blind |
| Everything is visible via an intuitive user interface | Users can see al relevant information such as:   * Cards * Chips * Dealer position/button * Pot size * Other players actions * Size of the blinds |
| User can see hand rankings | When a button is pressed a table of hand rankings pops up so that people can see how good their hand is/what they could achieve |

Design Section

**Decomposition of the problem**

1. Login Authentication system
   1. Input for username and password
   2. Hashing and salting of inputs
   3. Checks against server database
   4. Progresses to main menu if correct
   5. Can create a new username and password
2. Account Creator
   1. Ability to add
      1. Username
      2. Password
      3. Password verification (enter it twice)
   2. Block out password so user can’t see it as they type
   3. Creates and stores new Hash and Salt
3. Main menu buttons
   1. View server list
      1. A full refreshable and current list of available games with their status
      2. A back button
      3. The ability to join a game which opens a sperate window
   2. Host a game
      1. A tutorial on port forwarding
      2. A lobby with the option to start/cancel the game at any point
      3. You can see when people join
   3. View statistics
4. Gameplay
   1. An exit button
   2. Visual indicator of what your cards are
   3. Visual indicator of everyone’s balance
   4. Full deck of cards created
      1. Fully random hand shuffling
      2. Each card can only be used once
   5. Prompts to act on your turn
      1. Either call raise or fold
         1. Call. Match the current pot or if its too expensive put you all in
         2. Raise. Allow the user to increase the bet by a certain amount
         3. Fold. Allow the user to exit the game
         4. Record individual players contributions separately
      2. Bets loop until everyone is paid up or folded
      3. A timer that when complete automatically folds inactive players
      4. Calculate the winner
         1. Check cards for hands such as straight, flush etc…
         2. Rank hands from winner to looser
         3. Remove folded hands
         4. Calculate pot allocations
            1. Account for split pots by using the individual contribution and not allowing them to collect more than each player multiplied by this value
   6. Can see other players actions in real time
5. Sql reader
   1. Necessary functions
      1. Append a new value/record to the database as a tuple to prevent sql injection
      2. Replace/overwrite entries when given the new value and the one to be replaced
      3. Read and return a set of records as a 2d array when given the sql code to be executed
      4. Read and write encrypted entries for password security using asymmetrical encryption where a single server has access to the private key

**Structure**

The Program is split into several modules to take advantage of Python’s powerful modular design. This helps me perform isolated tests of individual systems, makes working on the project easier a I don’t have to comb through an extremely large file to find what I’m working on. It also helps with the decomposition of the problem as I can run different aspects in different files. Here you can see the file/module structure of the end user’s application, the right side of the tree represents if the user is hosting the game and the left side represents when the user has joined a game hosted elsewhere. “Client\_A” and “client\_B” handle the majority of the networking and communication during the runtime of the game. “Mainloop” holds the classes used to run the game and all the code associated with it. “Mainprogramclient” holds the user interface and menu code that allow the player to select what they want to do.

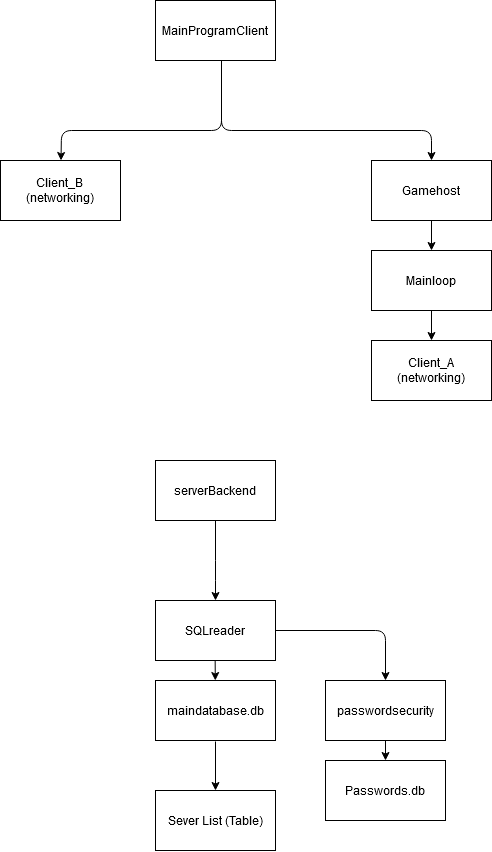


Figure 1 - The file/module structure of the application

Below is the module/file structure for the server. This server is hosted on my network and has all the databases needed for the game as well as handling connecting players together. I used an “SQLreader” module I wrote however it was not as powerful as intended because you can’t insert a variable as the table name which meant the functions could not be as general purpose and reusable as I would have liked and be heavily specialised to its use case.

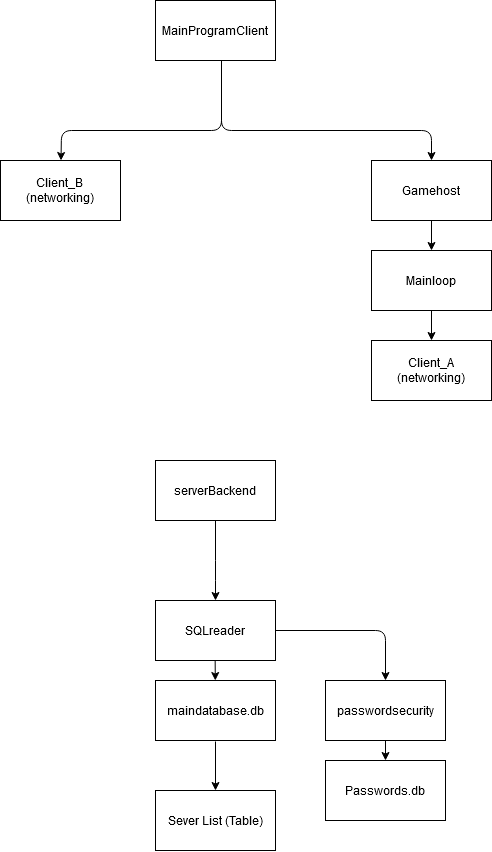


Figure 2 - The File/module structure of the server

Figure 3- The error message given when attempting to use variables as table names

**Choosing my resources**

**Choosing an encryption library and algorithm**

* I initially planned to use the “pycrypto” library as it seemed simple and lightweight while still being powerful enough to protect my users. However after some research I discovered that this library is unmaintained since 2013 and is therefore not safe for use. This prompted a switch to “cryptography” a more complex encryption library.
* For the purpose of sending sensitive fata to and from a downloadable piece of software a symmetrical encryption algorithm would not be sufficient meaning I had to use an asymmetrical algorithm. I settled on rsa due to its faster validation times and similar security (when using a 2048 character modulus) to dsa while being very well documented.
* Cryptography also supports all major methods of hashing and salting which are necessary for password security so that I am in accordance of the data protection act

**Selecting a database manager**

The database manager used is extremely important as you will spend a lot of time working on it so it must be easy for you to develop on while also being powerful enough to perform the tasks required of it. I decided to compare “mySQL” to “SQLite”, two very popular, python compatible database managers.

**mySQL**

* Industry standard, very well documented and supported
* Allows multiple simultaneous edits which may be useful but due to my limited use of the threading module its use case is limited
* Cross language making it useful as a skill to learn and be familiar with
* More advanced and flexible
* Inbuilt server capabilities

**SQLite**

* I am already intimately familiar with the SQLite browser and python module
* SQLite is, as the name implies, very lightweight and simple which is very useful with the time limitations and scope of this project
* I have already written useful code I can simply reuse
* I have to hack a server together using subpar techniques

Due to the tight time constraints of this project I elected to use SQLite as despite being worse than mySQL I am already very familiar with it and can reuse code/functions from previous projects whereas mySQL has a steep learning curve and would require significant time investment to learn. Furthermore all the functionality of mySQL is possible by combining modules such as socket, pickle and SQLite.

**Selecting a data serialiser**

To send data over a socket in python it must be the “bytes” datatype, for strings this is achieved using the .encode() method however for more complex objects such as arrays and dictionaries a data serialised is needed. I compared the two most popular solutions, json and pickle.

**JSON**

* Human readable
* Cross language (don’t plan on using multiple anyway)
* Industry standard, better documented and supported
* Much safer, as I’m passing ip addresses etc may be an advantage
* Can’t deal with tuples etc and makes non string keys in dictionaries into strings, this causes issues with my mainloop.py using integer keys on a dictionary

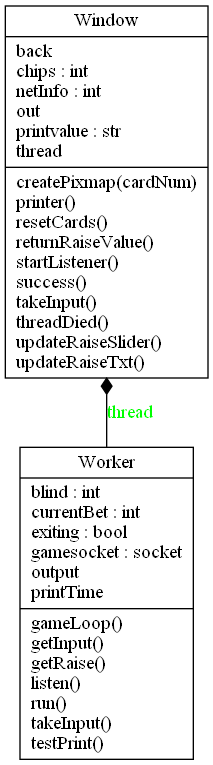
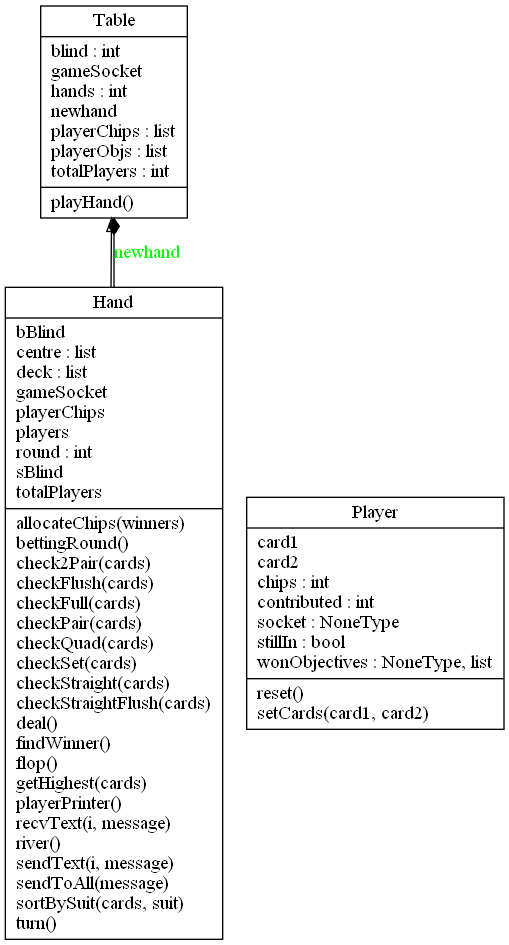
**Pickle**

* Faster
* Can send classes
* Allows for data serialization of more data types and objects and better support for pythonic use cases due to being built from the ground up for python

Note: Could be faster using cpickle but this is not supported in python 3.x yet

Due to the better python support for pickle I elected to use it for my data serialiser

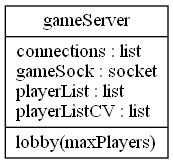
**UML Diagrams**



Client program

Game loop

Server Browser

****

|  |
| --- |
| Hand |
| Attributes:   * integer sBlind * integer bBlind * 2d-array deck * 1d-Array players * 2d-array centre * Integer round * 2d array connected |
| Methods:   * contructor * deal * sendtxt * recvtxt * bettinground * flop * turn * river * allocateChips * findWinner * checkFlush * checkStraight * checkStraightFlush * checkPair * check2pair * checkSet * checkQuad * checkFull * getHighest * sortBySuit |

|  |
| --- |
| Table |
| Attributes:   * TotalPlayers * playerChips * hands * blind |
| Methods:   * contructor * playHand |

|  |
| --- |
| player |
| Attributes:   * stillIn * contributed * wonObjectives * chips |
| Methods:   * contructor * setCards |

Development section