Pictionary

Chris Taormino Michaela Gates Judah Parham
Jordan Esty

December 16, 2015

Contents

Chapter 1

Background and Scope

1.1 Background

The end goal of this software engineering project is to create a game, Pictionary, that accurately captures the features and requirements listed by the client. The end result is to create the game, as named by the Client, Pictionary. The game Pictionary, will consist of four teams. The teams will consist of players that have signed up to play the game. The sign up process simply requires a unique combination of a username/password. Once the new player signs up, they can pick a team that they will represent when they play Pictionary. These teams are Ochre, Puce, Mauve and Blue. Players will be able to select one of the four teams on sign up. With each team chosen, the next player will not be able to select that same team in an effort to balance team sizes. The members of a team and the ranking of those members will be seen via the Leaderboard. The leaderboard has the overall score of each team, each member of the team and those members are organized by their contribution to the team. After signing up and selecting a team, players will be able to join games. In a game, at least two teams and two players must be present. Once a player wants to join a game, they will go to a ready up screen where they will confirm they are willing to participate in a game. The game client will randomly select a drawer. All the other people chosen will be assigned the role of guesser. The drawer will select a category and the game client will randomly select a word from that category. The drawer will then select a time limit on how long the guessing/drawing period will last. Once the game begins, the drawer will draw in the draw field and the guessers will input the guesses into the guess field. Each round players will be able to change their score. Depending on their performance, their score will increase or decrease. The ideal score is a lower score1. A bonus is a favorable additive on a users score for that round. A penalty is an increase in the users score. Bonuses and penalties are warranted based on certain actions. The first person to guess the drawing correctly will receive a bonus for being correct first. If time runs out and no one guesses correctly, the drawer receives a penalty. If everyone guesses correctly, the drawer receives a bonus. If the drawer does not start drawing within 15 seconds of the draw/guess phase starting, the round ends and the drawer is disconnected. If that particular user habitually disconnects, they will receive a penalty. In a typical course of events, only the person who guesses first will receive a bonus that round. Once the round ends, players will see their score for that round and be able to make a choice as to whether they will join for another round.

1.2 Scope

The developers are expected to create the game, Pictionary, to the specifications agreed upon by the customer and developers. The developers will handle the software engineering scope of the project and the customer will provide feedback on the final product and any prototypes produced for the customer. The developers are not responsible for faulty hardware that the customer may or may not possess. Faulty hardware can be defined as a computer that is not up to date with its Operating System, damaged hardware or attempting to execute the program outside of its intended use.

1.3 Stakeholders

1.3.1 Dr. Sibren Isaacman

Our customer is Dr. Sibren Isaacman. He has commissioned Group 3 Incorporated LLC to design, create and maintain a game named Pictionary. After several meetings with Dr. Isaacman, we elicited his requirements for the program. Once we confirmed his requirements and specifications, Group 3 set out to create Pictionary.

1.3.2 Group 3 LLC

Group 3 Inc. LLCs goal is to create, test and maintain Pictionary based on the requirements and specifications listed by the Customer. Creation includes the requirements/specification elicitation, design and coding. Testing and maintaining the code is the part of the development process that will take up the most time. This includes, but is not limited to, addressing bugs, adding features, and ensuring the games playability.

1.3.3 Potential Players

Players are the people that will interact with the program Pictionary. They are assigned to one of two roles based on how they connect to a game. These roles are Drawer and Guesser.

- Player - Drawer: Players that connect to a new game session first are drawers. The drawers goal is to ensure that the word selected is guessed correctly and quickly based on their ability to draw the word. The drawer ideally wants everyone to guess their drawing correctly. If no one guesses the word based on the drawers ability to represent it, the drawer is penalized.

- Player - Guesser: The role of guesser is to correctly and quickly guess the word being represented by the drawer. If the guesser is the first person to answer correctly, they receive a bonus. All subsequent correct guesses are not rewarded with a bonus. Any guesser that does not answer correctly before the time runs out will be penalized.

Chapter 2

Requirements

2.1 Use Case Diagram

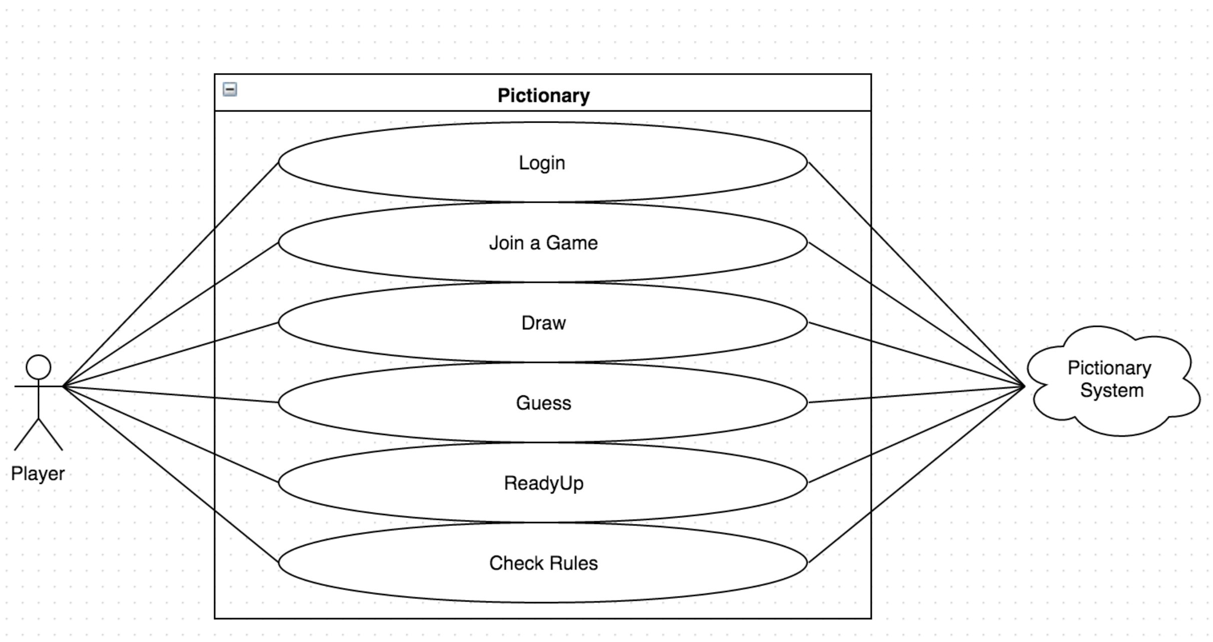


Figure 2.1: Use Case Diagram of the system.

2.2 Features

The following are the features of our software project, listed from most important to least important:

1. Login/Logout
    (a) This includes users information being saved and rewritten properly
2. Game Start
    (a) Ready up (timer)
    (b) Category/Time selection
        i. Word Assignment
3. Drawing/Guessing
4. Round Rotation
5. Leaderboard/Scores

2.3 Use Cases

2.3.1 First time User Login - UC01

Primary Actor: New Player (First-Time User)
Preconditions: User cannot choose a name that is already in use in the list of names already registered
Success Guarantee: The new user now has an associated account, and can properly log-in to the game
Main Success Scenario:

1. User will select to register as a new player
2. They will be prompted to insert a user-name.
3. A unique name will result in the game prompting for a password
4. The user will be created and registered.
5. All values related to the score of the user will be set to their respective default states
6. Team selection - Each time a new user is added, the team they choose will not be available for selection. This is in an effort to balance teams

Extensions: If the entered user-name is already invalid - already taken, longer than 16 characters, is not alphanumeric. Error will be described and user shall then reenter information.
Special Requirements: Internet access for the user
Open Issues: None

2.3.2 Login - UC02

Primary Actor: User (Returning User Login)
Preconditions: The user successfully logs in with their correct combination of username/password
Success Guarantee: Registered User has successfully logged into the game
Main Success Scenario:

1. Registered User opens software
2. Registered User attempts login
3. Server authenticates logins from Registered User
4. Registered User successfully logs in
5. Registered User now sees the main menu

Extensions:

- If Registered User enters incorrect login information, deny login.
- If Player has not yet become a Registered User, deny login.

Special Requirements:

- Established Internet connection between Registered User and Server
- Player is a Registered User

Open Issues: None

2.3.3 Player Joins a Game - UC03

Primary Actor: Player (Ready Up)
Preconditions: Registered User has logged in and desires to join a game
Success Guarantee: First Registered User has become the host of a new game. The following Registered Users to join a game will join this original User.
Main Success Scenario:

1. Registered User selects to join a game
2. Registered User is asked if they are ready for a game to start - there is a confirm or decline option
3. On confirm, the Registered User is assigned a role
4. On decline, the Registered User is removed from the queue

Extensions:

- If there were only two Registered Users are in queue and one leaves, the game is ended

- If only one team readies up, the game is not valid because two teams are not represented

Special Requirements: Established Internet connection between the Host and Server
Open Issues: None

2.3.4 Drawer Category Selection - UC04

Primary Actor: Host
Preconditions: There must be categories to choose from and time is selected by radio buttons. A drawer must have been chosen; drawer must be logged in; must be in a game.
Success Guarantee: A Registered User has become a drawer, who then selects the category
Main Success Scenario:

1. A Registered User is selected to be a Drawer
2. The Drawer chooses the category and selects the time
3. A word is assigned to the drawer from that category
4. The round starts

Extensions:

- A time limit to chose a time per round should be imposed to ensure that either a time is selected or the round ends so that the selection process can be redone.

Special Requirements: Established Internet connection between the Host and Server
Open Issues: None

2.3.5 Drawing in a Game - UC05

Primary Actor: Drawer (Drawing)
Preconditions: A round must be started, category and time must have been selected, word must be assigned.
Success Guarantee: A Registered User has become a Drawer. The game has started, and drawing begins.
Main Success Scenario:

1. User selects a color and clicks on the drawing space
2. On click, the line begins to draw
3. On release, the program stops drawing the line and the program updates the drawing to the server

4. The other guessers receive the updated picture on their screens

Extensions:

- If there is a loss of connection or break in the server-client, the round will end.

Special Requirements: Established Internet connection between the Registered Users and Server
Open Issues: None

2.3.6 Guessing by Player - UC06

Primary Actor: Guesser (Guessing)
Preconditions: A round must be started, category and time must have been selected, word must be assigned. Display field must be functional.
Success Guarantee: The Guesser has viewed the drawing and has successfully made a guess.
Main Success Scenario:

1. Drawer has seen the in-progress drawing created by the Drawer
2. The Drawer enters their guess in the field
3. The game checks if the response is correct
4. Correct answers result in a win
5. Incorrect answers result in a negative impact on scores

Extensions:

- The game should always accept and be able to compare the answer provided.
- Unless there is a network error, the guess and check function should not fail.

Special Requirements: Established Internet connection between the Players and Server
Open Issues: None

2.3.7 New Round - UC07

Primary Actor: Interface (Round Rotation)
Preconditions: The players must accept the ready up invitation. At least one member from two teams will accept the invitation to a new round. A round must have ended.
Success Guarantee:
Main Success Scenario:

1. At least one member from two teams will accept the invitation to a new round
2. They will choose to confirm their participation in the next round or decline and be removed from the selection queue
3. A new round will begin

Extensions:

- Less than two people or less than two teams are present

Special Requirements: Established Internet connection between the Players and Server
Open Issues: None

2.4 Domain Model

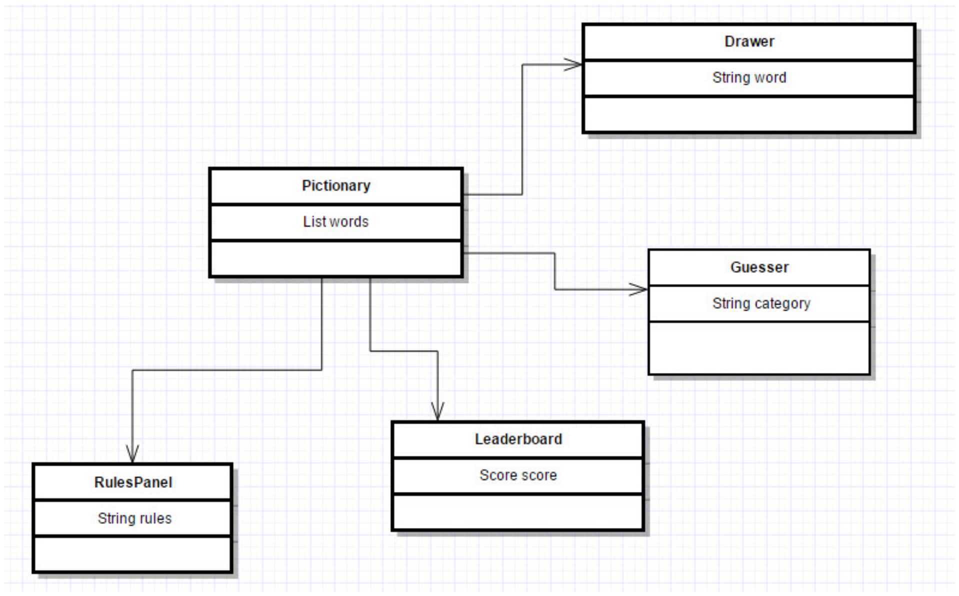


Figure 2.2: Domain Model of the System

2.5 Non-Functional Requirements

2.5.1 Technology Requirements

- The game shall be functional on supported Windows versions
- The game shall be functional on supported Apple versions

- The game shall support a minimum resolution of 800x600
- The game shall require a minimum of 100 MB hard disk space
- The game shall use the Oracle DB to hold data
- The game shall require the user to have ports 9090, and 9091 open
- The game shall use Java's built in socket layer
- The game shall require a minimum Java runtime environment of 1.7
- The game shall require a minimum of 1.5 GB of RAM
- The game shall require a sound card
- The game shall allow the user to interface using a mouse and keyboard
- The game shall require at a minimum the Pentium II Intel chipset

2.5.2 Requirements from External Systems

- The game shall require a minimum Java Runtime Environment of 1.7
- The game shall be browser independent. The browsers which will be tested for compatibility are; Chrome, Internet Explorer, Edge, Firefox, and Safari.

2.5.3 Performance and Scalability

- The game shall support up to 16 players
- The game shall respond within a second to a users command

2.5.4 Use and Usability

The system takes into consideration the following accessibility features:

- The game shall require a minimum of 2 players representing two teams before starting an instance of the game
- The game shall ask each user before starting a new instance of the game, whether they want to play or not
- The game shall ask for confirmation before starting, joining, or leaving the game
- The game shall kick drawing users after a half a minute of inactivity
- The game shall require a username and password for each user
- The game shall require the user to login each time the game is started
- The game shall keep track of statistics for each user. These statistics include; total points, correct guesses, games won and team

2.5.5 Maintenance and Portability

- The game shall be compatible with any Java Runtime Environment of version 1.8 or newer
- Game allows Admin to add/remove words

Chapter 3

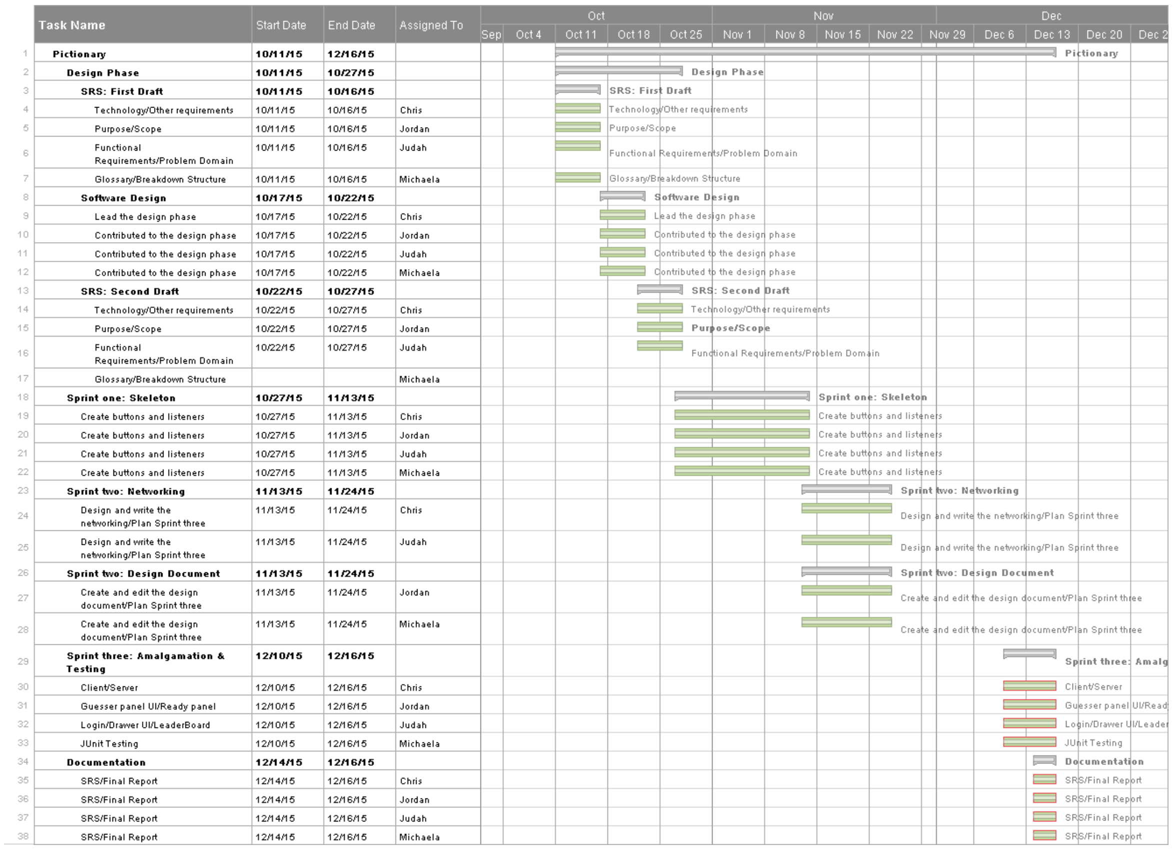
Plan of Development

3.1 Development Process

Group 3 Inc. LLC plans to implement a SCRUM based means of software development process. Our development team shall meet at least once a week on Tuesdays throughout the developmental process. There will be three main sprints that will encompass the completion of the game Pictionary. At the end of each sprint, a main component will be completed and tested. We will program in steps to ensure that the game is completed in its whole by the end of the third sprint. Our documentation techniques will be a waterfall documentation. We will work our way from the top level, where we establish terms and standards in an abstract sense, and work our way down to the final product in a more definite and final form.

3.2 Gnatt Chart

It proved rather difficult to estimate how much time we would need to complete this project. The main sources of difficulty came from varying schedules, other work from other classes and the complexity of the project. We would rarely project the time required correctly and sometimes, the correct amount of time needed was not always a feasible time given each member's personal workload. Chris was planned for a total of 126 hours. In total, he spent 128 person hours. Jordan was planned for a total of 68 hours. In total, he spent 67 hours. Judah was planned for a total of 66 person hours. In total, he spent 79 person hours. Michaela was planned for a total of 68 person hours. In total, she spent 72 hours.



	Task Name	Start Date	End Date	Assigned To
1	**Pictionary**	**10/11/15**	**12/16/15**	
2	**Design Phase**	**10/11/15**	**10/27/15**	
3	**SRS: First Draft**	**10/11/15**	**10/16/15**	
4	Technology/Other requirements	10/11/15	10/16/15	Chris
5	Purpose/Scope	10/11/15	10/16/15	Jordan
6	Functional Requirements/Problem Domain	10/11/15	10/16/15	Judah
7	Glossary/Breakdown Structure	10/11/15	10/16/15	Michaela
8	**Software Design**	**10/17/15**	**10/22/15**	
9	Lead the design phase	10/17/15	10/22/15	Chris
10	Contributed to the design phase	10/17/15	10/22/15	Jordan
11	Contributed to the design phase	10/17/15	10/22/15	Judah
12	Contributed to the design phase	10/17/15	10/22/15	Michaela
13	**SRS: Second Draft**	**10/22/15**	**10/27/15**	
14	Technology/Other requirements	10/22/15	10/27/15	Chris
15	Purpose/Scope	10/22/15	10/27/15	Jordan
16	Functional Requirements/Problem Domain	10/22/15	10/27/15	Judah
17	Glossary/Breakdown Structure			Michaela
18	**Sprint one: Skeleton**	**10/27/15**	**11/13/15**	
19	Create buttons and listeners	10/27/15	11/13/15	Chris
20	Create buttons and listeners	10/27/15	11/13/15	Jordan
21	Create buttons and listeners	10/27/15	11/13/15	Judah
22	Create buttons and listeners	10/27/15	11/13/15	Michaela
23	**Sprint two: Networking**	**11/13/15**	**11/24/15**	
24	Design and write the networking/Plan Sprint three	11/13/15	11/24/15	Chris
25	Design and write the networking/Plan Sprint three	11/13/15	11/24/15	Judah
26	**Sprint two: Design Document**	**11/13/15**	**11/24/15**	
27	Create and edit the design document/Plan Sprint three	11/13/15	11/24/15	Jordan
28	Create and edit the design document/Plan Sprint three	11/13/15	11/24/15	Michaela
29	**Sprint three: Amalgamation & Testing**	**12/10/15**	**12/16/15**	
30	Client/Server	12/10/15	12/16/15	Chris
31	Guesser panel UI/Ready panel	12/10/15	12/16/15	Jordan
32	Login/Drawer UI/LeaderBoard	12/10/15	12/16/15	Judah
33	JUnit Testing	12/10/15	12/16/15	Michaela
34	**Documentation**	**12/14/15**	**12/16/15**	
35	SRS/Final Report	12/14/15	12/16/15	Chris
36	SRS/Final Report	12/14/15	12/16/15	Jordan
37	SRS/Final Report	12/14/15	12/16/15	Judah
38	SRS/Final Report	12/14/15	12/16/15	Michaela

Figure 3.1: Gantt Chart of our Progress

3.3 Team's Weekly Effort

3.3.1 Chris Taormino

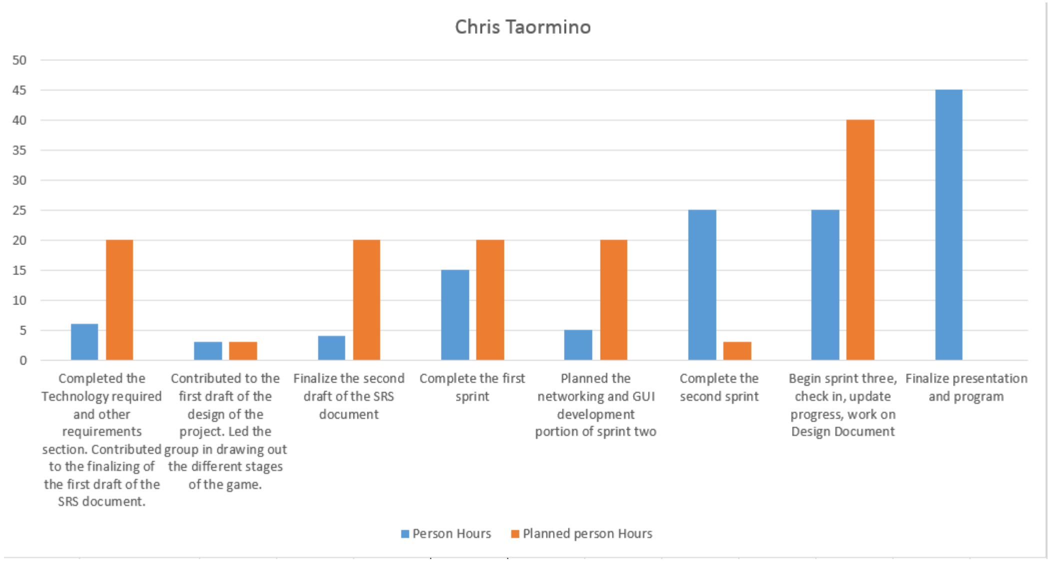


Figure 3.2: Chris's Weekly Effort

3.3.2 Michaela Gates

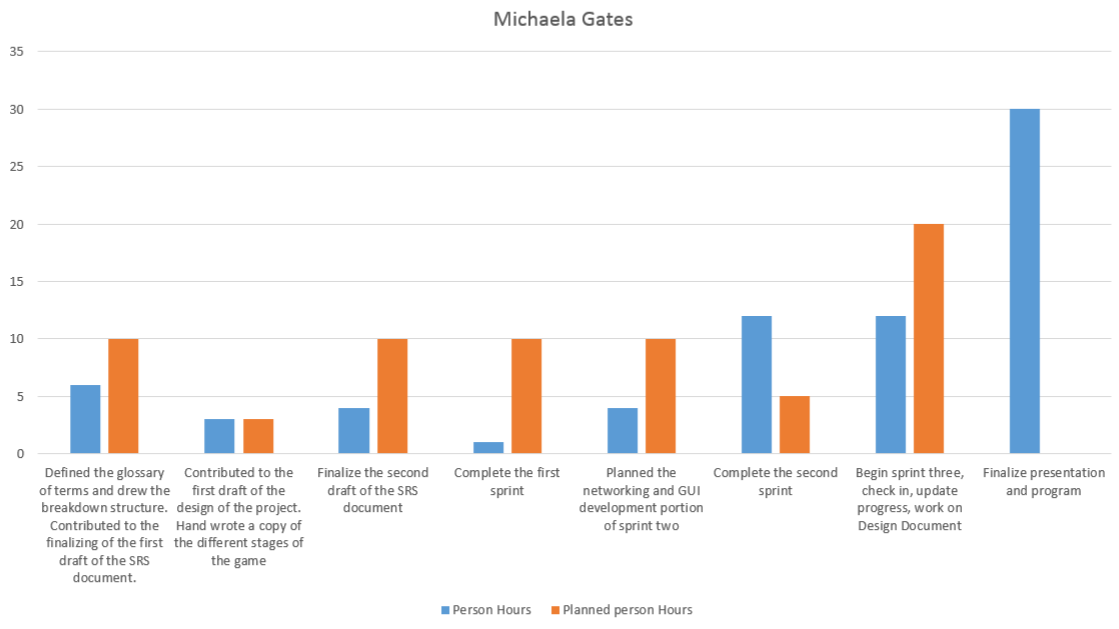


Figure 3.3: Michaela's Weekly Effort

3.3.3 Judah Parham

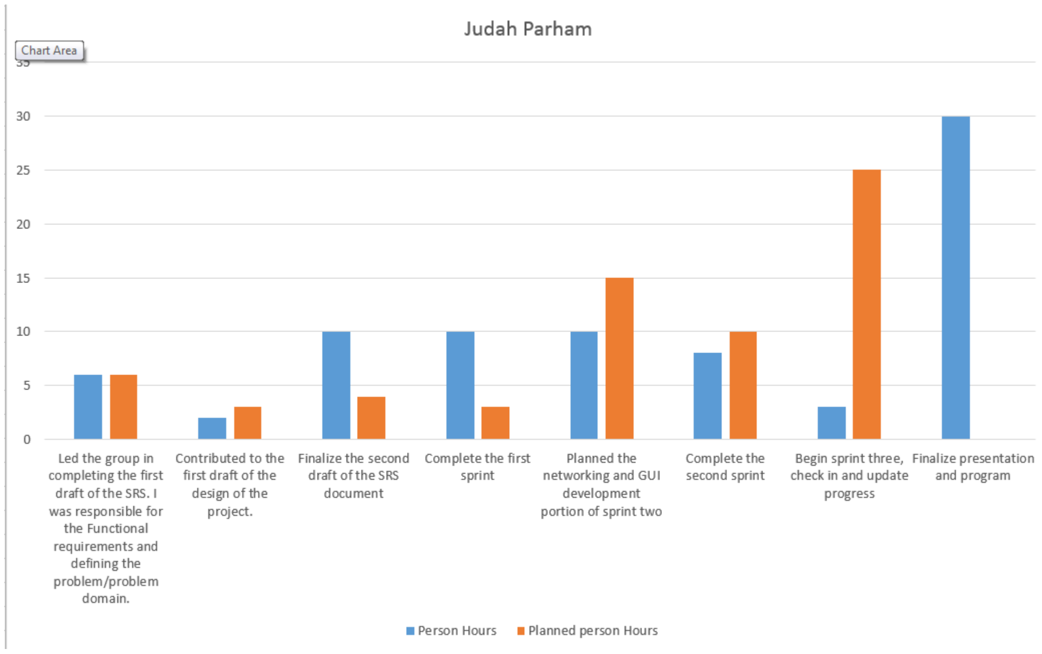


Figure 3.4: Judah's Weekly Effort

3.3.4 Jordan Esty

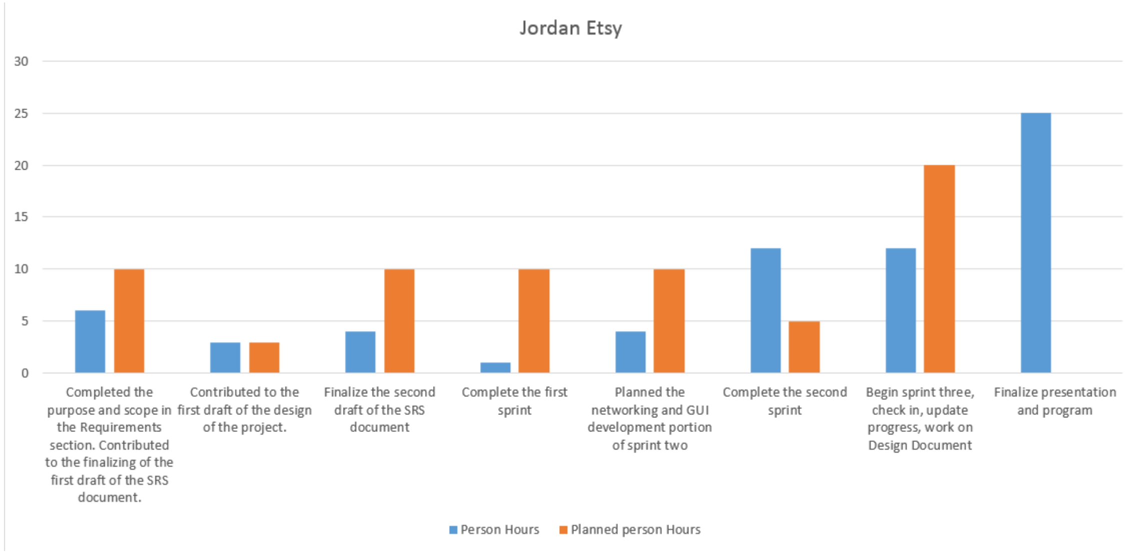


Figure 3.5: Jordan's Weekly Effort

3.4 Line Diagrams

3.4.1 Chris Taormino

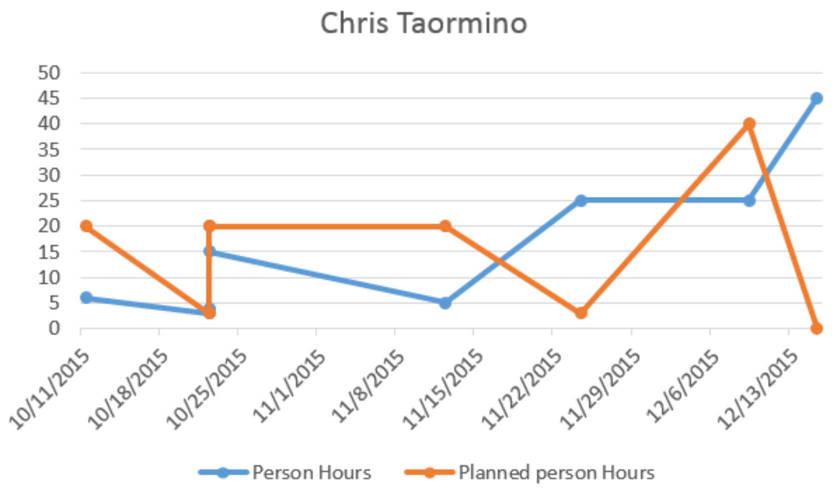


Figure 3.6: Line Diagram of Chris's Man Hours

3.4.2 Michaela Gates

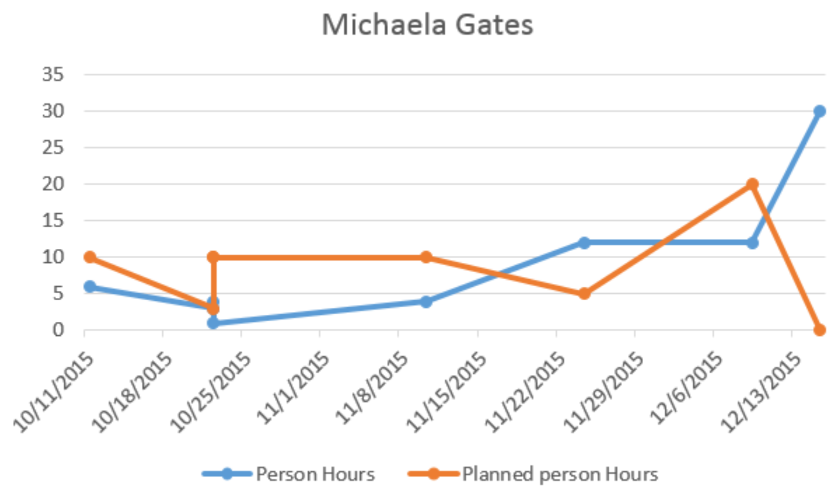


Figure 3.7: Line Diagram of Michaela's Man Hours

3.4.3 Judah Parham

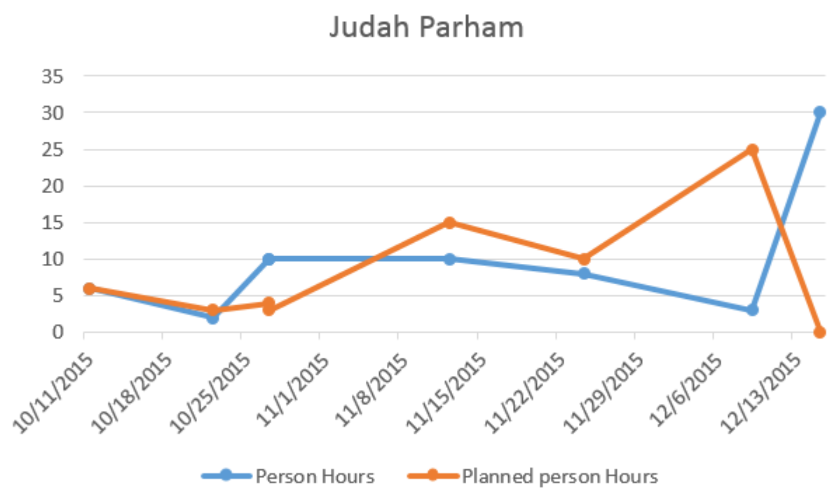


Figure 3.8: Line Diagram of Judah's Man Hours

3.4.4 Jordan Esty

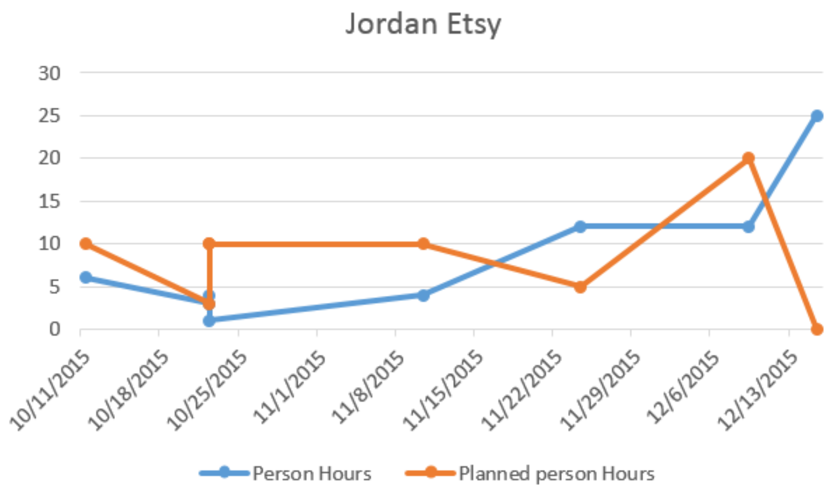


Figure 3.9: Line Diagram of Jordan's Man Hours

Chapter 4

Design Document

4.1 Software Architecture

This system uses a combination of Client-Server and Repository Architectures. In regards to the system having aspects of a Repository, the system retrieves information from the Database which has user and game information stored. In regards to the system having aspects of Client-Server, the end user communicates with the Pictionary system by requesting games and sending guesses to it. The system sends picture created by the user designated as a drawer and other game related information to the end user, if they are designated as a guesser. The end user is whomever is a current player when the server is sending out game information.

4.2 Description of Components

- Pictionary System: Interacts with and exchanges information among all the panels and the database
- Title Panel: Appears when the game is initialized
- Login Panel: Appears after the Title Panel. Allows users to log in using their credentials. Gives users the option of creating an account if they do not already have one. Information entered in this panel is saved in and checked against information within the database.
- CreateAccountPanel: As with the Login Panel, information entered in the Panel is exchanged with the database.
- MenuPanel: Interacts with the database to retrieve information about the current user. User communicates with Pictionary System with regards to initializing or joining a game.

- DrawerPanel: Pictionary System designates one person this panel and sends updates on the drawings
- GuesserPanel: Pictionary sends updates to this panel which include drawer and time updates, as well as. This panel sends guesses to the Pictionary System and an update as to whether a sent guess is correct is returned.
- ReadyPanel: Before every game starts, players either decide to go back to the menu or join the game, interacting with the Menu Panel and the Pictionary System
- Leaderboard Panel: Interacts with the database to retrieve game rankings
- Information Database: Interacts with the Pictionary System to retrieve user and game information.

4.3 Class Diagram

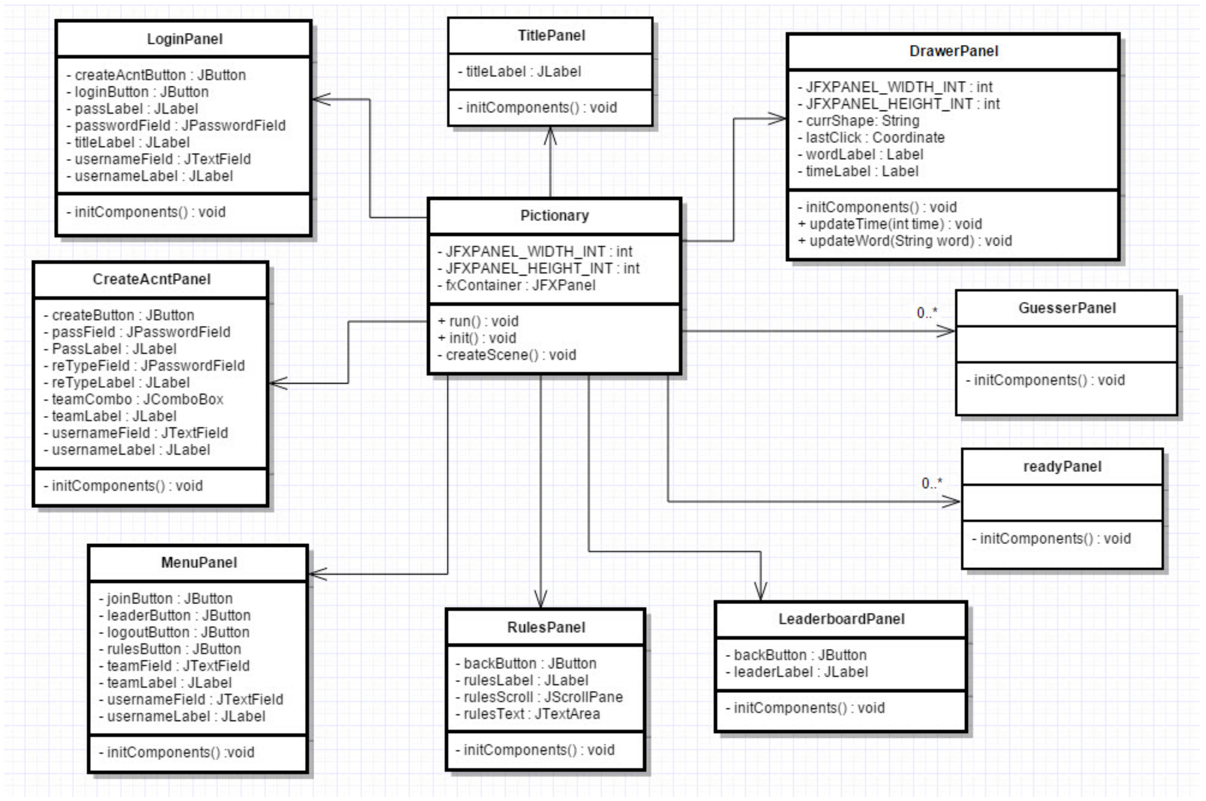


Figure 4.1: UML Class Diagram of the System

4.4 Sequence Diagram

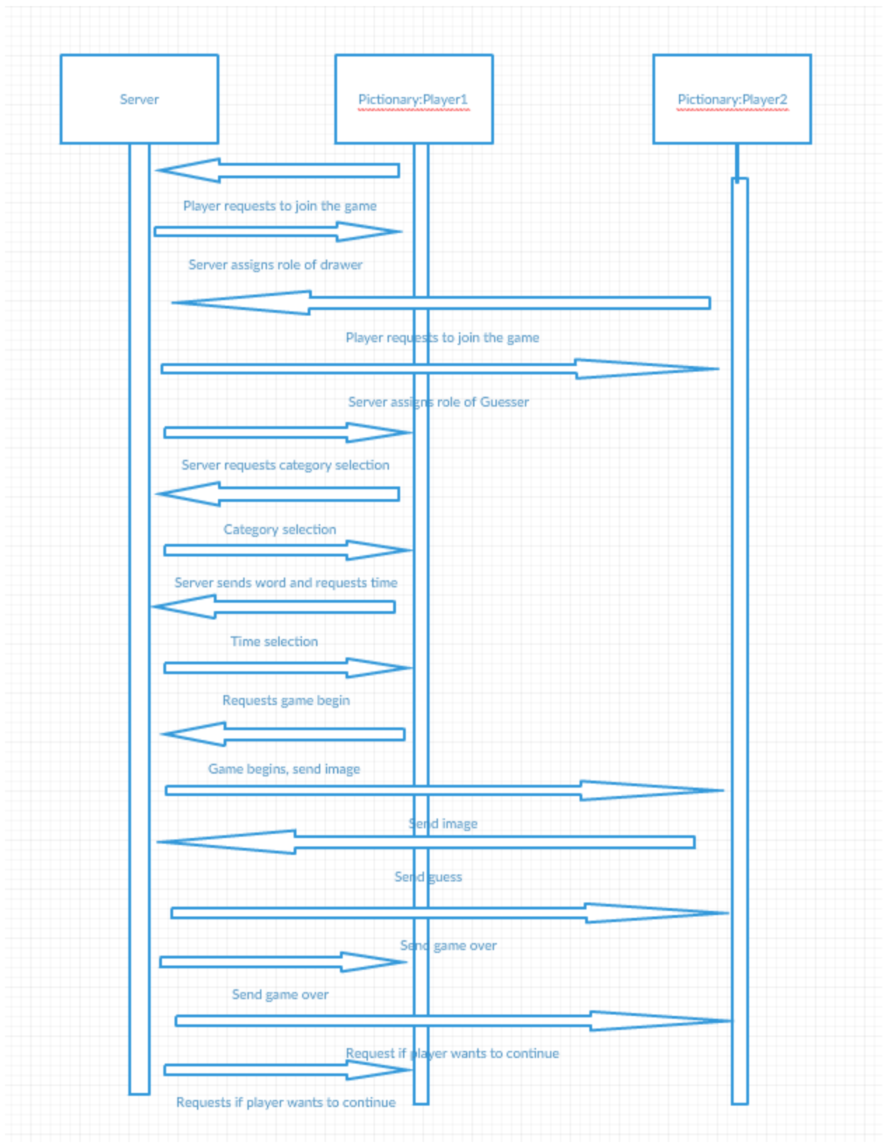

Server
Pictionary:Player1
Pictionary:Player2
Player requests to join the game
Server assigns role of drawer
Player requests to join the game
Server assigns role of Guesser
Server requests category selection
Category selection
Server sends word and requests time
Time selection
Requests game begin
Game begins, send image
Send image
Send guess
Send game over
Send game over
Request if player wants to continue
Requests if player wants to continue

Figure 4.2: Sequence Diagram of Guesser guessing correctly on first try

4.5 State Diagram

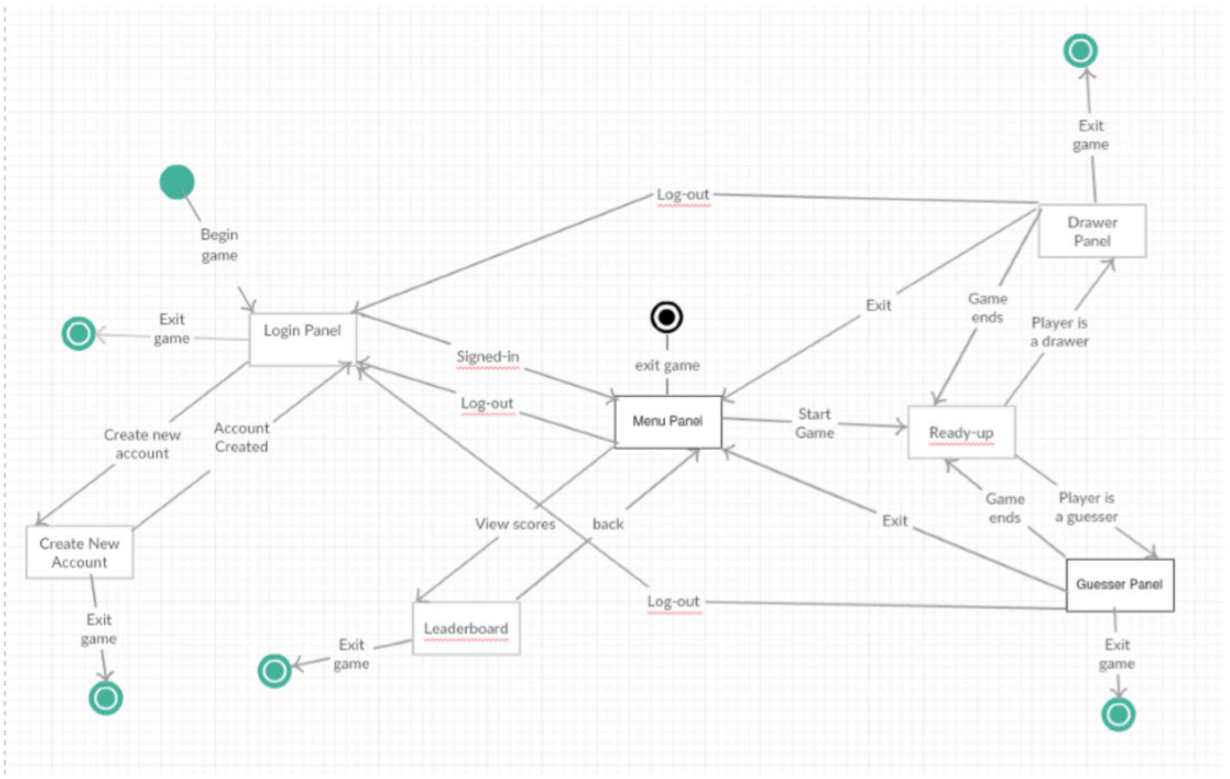


Figure 4.3: State Diagram of GUI

4.6 ER Diagram

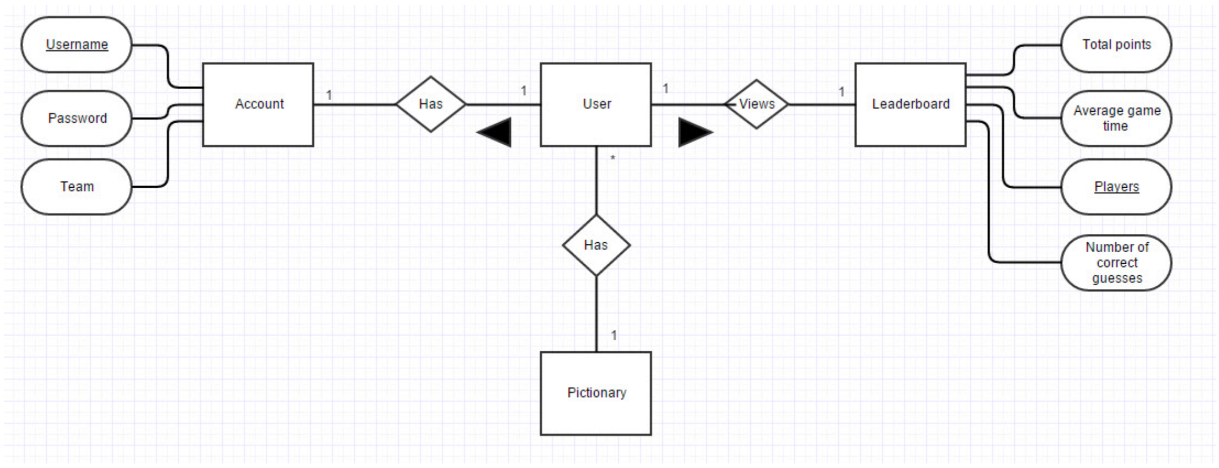


Figure 4.4: ER Diagram of the System

4.7 Design Architecture

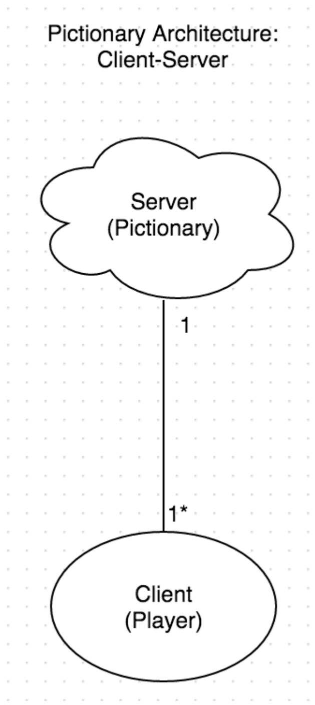


Figure 4.5: Design of the Database

Chapter 5

Test Plan and Testing

5.1 Test Plan

We planned to test by using an instance of Robot class to click through the GUI's, enter text and check the output of those actions against our desired output. For example, the program would start and the game would display the login menu. The Robot, bot, would then enter a predetermined user name and password combination. The bot would then click login. Bot would test cases where the combination would be correct, the username is wrong, the password is wrong and when both the username and password are wrong. Given the expected output for these inputs, we can determine whether the code works properly or not.

5.2 Code Coverage

The use of the Robot class allows us to navigate the GUI's depending on the input given to the Robot. It was difficult to get the robot to click in the right areas.Whenever the Robot clicks or enters text, some lines of code are executed based on that action. The parts that involved correct inputs were the easiest to test and yielded the highest coverage. When we tried to test the code based on bad inputs, the code became a bit more difficult to handle. It proved difficult to test exceptions that were thrown and cases would not fully or properly execute.

5.3 Integration Testing

We did not test exceptions which reduces the amount of coverage we could have obtained. If we had tested exceptions, we would have been able to achieve much higher code coverage.

Chapter 6

User Manual

6.1 Introduction

Pictionary is a multiplayer, word guessing game for two to sixteen players; One player is assigned a word that they will attempt to draw; All other players will attempt to guess the word as it is drawn.

6.2 Login/Create Account

New players should create an account if they want to play the game. When creating a new account, players will pick a user-name, password and team. Once these options have been chosen, they can log in like normal. Returning players should use their user-name and password combination. Players who have forgotten their user-name or password will unfortunately have to create a new account.

6.3 Check Leaderboard

The leader board will display the rankings of each team and the players on each team. The teams will be divided by their names and players will be organized based on their score.

6.4 Check Rules

The user will be able to check the rules. These should be read by each user before they begin to play the game so that they have a firm grasp of the flow of the game.

6.5 Join Game

If there is no game available and at least one person from two teams are represented, the two players will join a game. If less than two teams are represented, the game will not start. If the game starts and one of the members leaves resulting in a misrepresentation, the game will end.

6.6 Drawer

If the user is the first person to connect to the game, they will be assigned the role of drawer. The drawer will select a category and the game will select a random word from that category. Once the word is selected, the Drawer will choose a time limit. The time should be long enough such that the drawer has enough time to draw the object, but not so long that the guessers exhaust all of their guesses and have nothing to do.

6.7 Guesser

If the user is not the first person to connect to a new game, they are assigned the role of guesser. The goal for the guesser is to determine the word based on the drawing provided by the drawer and category reminder. The guesser who correctly answers first, receives a bonus for doing so. Subsequent correct guesses are not given any special rewards. Those who do not guess correctly in the allotted time frame are penalized.

6.8 Ready Up

After a round is completed, the users can select to either play again or exit the game. The game will then make sure that at least two teams are represented and will exclusively random select the next drawer. Once the drawer has been selected, the process begins again. In the event that less than two teams are represented in the next round, the game ends. Once a new set of players try to start a game with the appropriate amount of players, the game can begin again.

6.9 Exit/Logout

The user may exit the program at anytime. However, if the user exits in the middle of the game, habitually, the user will be penalized. If the user simply exits the game without logging out properly, they run the risk of losing whatever points they have accumulated, if any. If the user chooses to logout, they are able to do so without problem. The game will save their progress so far and exit the game safely.

6.10 Administrator

The administrator has the ability to change the list of available words. They have a unique login that is strictly reserved for them. The administrator cannot play the game.

Chapter 7

Reflection

7.1 The Software Process

Our group learned about working together as a team in order to complete the Pictionary project. Using SCRUM, it was easier to divide the various components of the project into more manageable time slots, although time management was also a topic that we learned a great deal about. We split up the components that needed to be done for each sprint during our weekly meetings on Tuesdays, where we also discussed our progress made that week, along with the assigned work for the next meeting. We learned how important it was to stay on schedule and not fall behind, as even a little bit would affect the next sprint. Overall, we learned how frustrating it can be at times to work in a group, but how rewarding it can be as well.

7.2 Teamwork

Our group learned about working together as a team throughout the Pictionary project development. We quickly learned how vital it was to split up the work efficiently, taking into consideration the strengths of each group member. During our meetings, we would be honest about our capabilities in order to get the most done. We would also present helpful feedback to one another, which made our work better as it forced us to think about one another's ideas and suggestions. Additionally, we learned some of the downsides to working in a team. Originally, it was somewhat difficult to find a meeting time in which everyone would be available. Every member also has to collectively agree on every decision concerning the developmental process.

7.3 Client and Requirements

Our experience with a pseudo real client, Dr. Isaacman, was a beneficial learning experience. We learned to professionally e-mail, meet with, and present work to him. Our meetings with the client were especially helpful in assuring our program met his requirements. When collecting requirements, we made sure to cover all possible cases from interfaces to game logic. Having iterations to present our progress to Dr. Isaacman helped us correct any requirement errors we may have had, and ensure that our progress was to his liking.

7.4 Software Design

Our experience with the process of designing software at the architectural, server-client, level went well despite the difficulty we faced. Looking back, there are many changes we would implement which would make the whole process run smoother. We used SCRUM as our development process, including three iterations being presented to our client. At the detailed design level, we used JavaDocs and JUnit testing to view the amount of code coverage we received from our unit tests.

7.5 Tools Used

The IDE used was mainly Eclipse, since it was easy for our group to edit the code in the repository through Eclipse. Each member also used their preferred diagramming tools, those being Gliffy and ArgoUML. When working with the documentation our group used Google Docs, Word Online, email and text messages to communicate back and forth.

7.6 Extensibility and Maintainability

Our software is extensible due to its modularity. It would be easy to add new interfaces, as we used JavaFX. Our code will be maintainable due to our Java Doc API and our JUnit tests. We were able to document a large amount of our code, so maintaining or updating in the future should not be an issue.

Chapter 8

API Documentation

8.1 Link to External API

The full API for this system may be found here: `http://www.cs.loyola.edu/~jesty/doc/index.html`

Appendix A

Glossary of Terms

A.1 Game Terms

A.1.1 Round

One complete cycle that includes a drawing and guessing period

A.1.2 Team

A group of users identified under one name; team names will be ochre, mauve, puce, and blue

A.1.3 Role

The objective of a player; will be different for drawers and guessers

A.1.4 Word

The noun or phrase in which the players are interacting with

A.1.5 Category

A division of words that have shared characteristics

A.1.6 Username

The name that is associated with a particular user; how each user logs in

A.1.7 Win

Correctly guessing a word

A.2 Technical Terms

A.2.1 Database

Stores login information and leaderboard information, located on the server

A.2.2 Host

Player who initiates the game

A.2.3 Multiplayer

Two to sixteen computers within the network, connected to a server within a game

A.2.4 Network

Collection of computers currently connected to the server

A.2.5 User

A person who is interacting with the game interaction in any capacity

A.2.6 Guesser

A player, who after a round is started, has the objective of guessing the drawn word correctly

A.2.7 Drawer

A player, who after a round is started, has the objective of drawing a depiction of the word the guessers have to guess

A.2.8 Player

A user who is interacting with the game application after the log-in prompt

A.2.9 New Player

A player that has not created login information

A.2.10 Registered User

A player who has created login information for this software

A.2.11 Server

Private computer that is hosting the game

A.2.12 Leaderboard

A table that shows each user with their respective teams, and all scores associated with each user

Appendix B

Table of Figures

B.1 Use Case Diagram

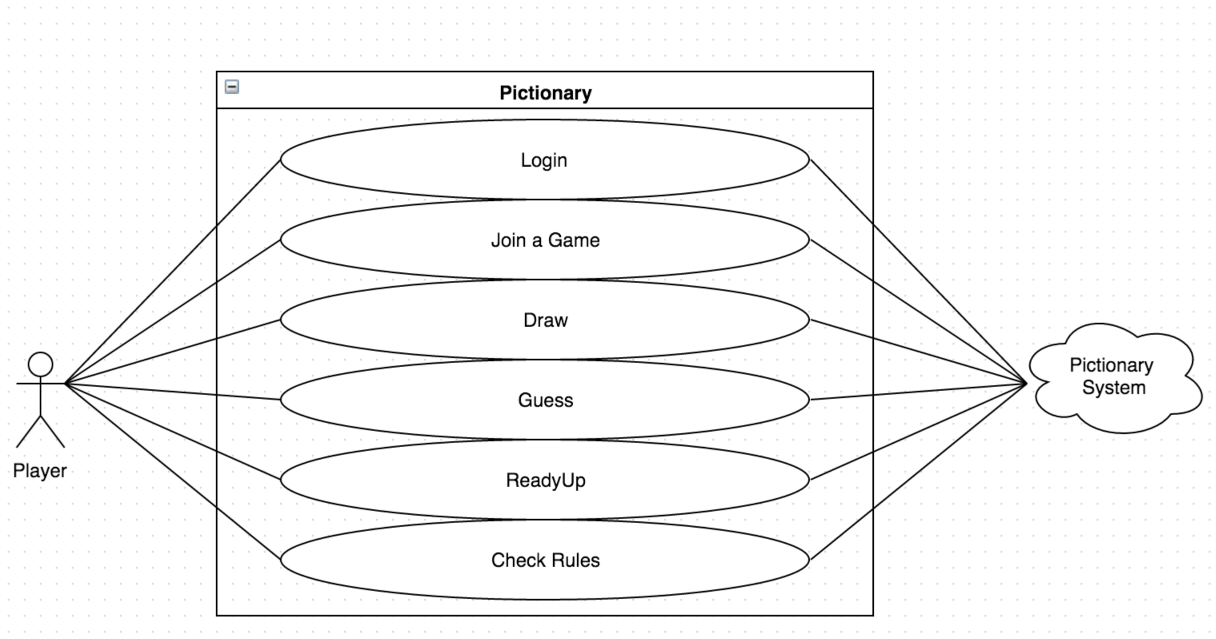


Figure B.1: Use Case Diagram of the System

B.2 Domain Model

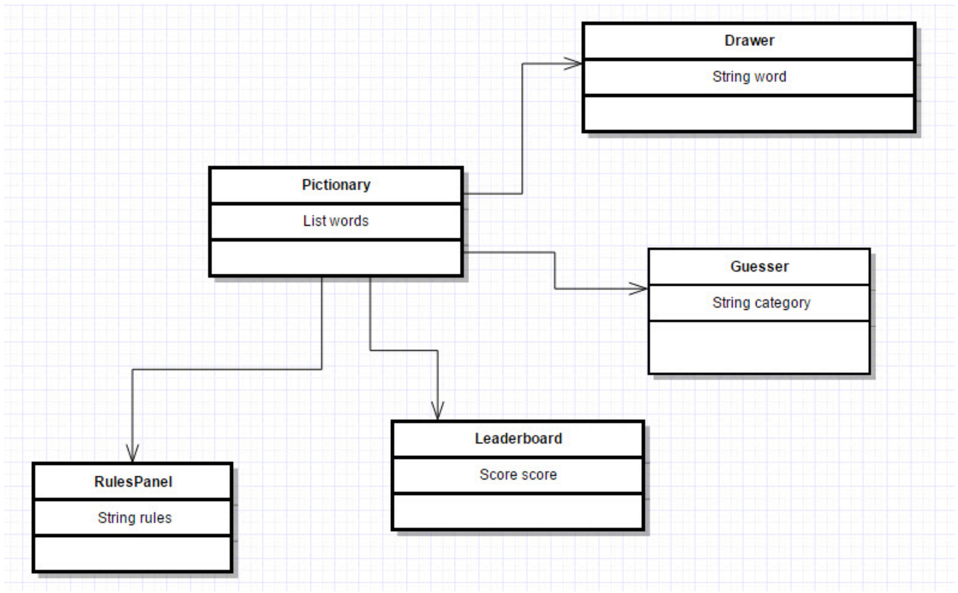
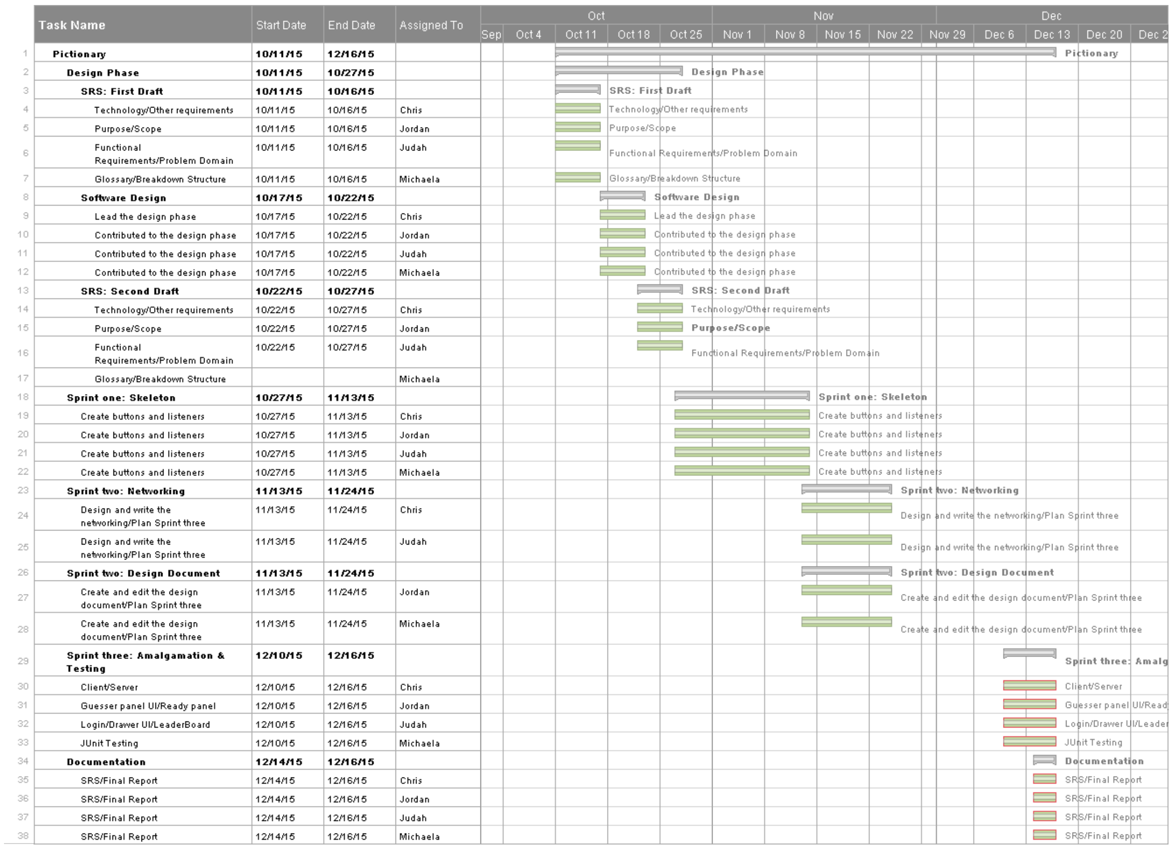


Figure B.2: Domain Model of the System

B.3 Gnatt Chart



	Task Name	Start Date	End Date	Assigned To
1	**Pictionary**	**10/11/15**	**12/16/15**	
2	**Design Phase**	**10/11/15**	**10/27/15**	
3	**SRS: First Draft**	**10/11/15**	**10/16/15**	
4	Technology/Other requirements	10/11/15	10/16/15	Chris
5	Purpose/Scope	10/11/15	10/16/15	Jordan
6	Functional Requirements/Problem Domain	10/11/15	10/16/15	Judah
7	Glossary/Breakdown Structure	10/11/15	10/16/15	Michaela
8	**Software Design**	**10/17/15**	**10/22/15**	
9	Lead the design phase	10/17/15	10/22/15	Chris
10	Contributed to the design phase	10/17/15	10/22/15	Jordan
11	Contributed to the design phase	10/17/15	10/22/15	Judah
12	Contributed to the design phase	10/17/15	10/22/15	Michaela
13	**SRS: Second Draft**	**10/22/15**	**10/27/15**	
14	Technology/Other requirements	10/22/15	10/27/15	Chris
15	Purpose/Scope	10/22/15	10/27/15	Jordan
16	Functional Requirements/Problem Domain	10/22/15	10/27/15	Judah
17	Glossary/Breakdown Structure			Michaela
18	**Sprint one: Skeleton**	**10/27/15**	**11/13/15**	
19	Create buttons and listeners	10/27/15	11/13/15	Chris
20	Create buttons and listeners	10/27/15	11/13/15	Jordan
21	Create buttons and listeners	10/27/15	11/13/15	Judah
22	Create buttons and listeners	10/27/15	11/13/15	Michaela
23	**Sprint two: Networking**	**11/13/15**	**11/24/15**	
24	Design and write the networking/Plan Sprint three	11/13/15	11/24/15	Chris
25	Design and write the networking/Plan Sprint three	11/13/15	11/24/15	Judah
26	**Sprint two: Design Document**	**11/13/15**	**11/24/15**	
27	Create and edit the design document/Plan Sprint three	11/13/15	11/24/15	Jordan
28	Create and edit the design document/Plan Sprint three	11/13/15	11/24/15	Michaela
29	**Sprint three: Amalgamation & Testing**	**12/10/15**	**12/16/15**	
30	Client/Server	12/10/15	12/16/15	Chris
31	Guesser panel UI/Ready panel	12/10/15	12/16/15	Jordan
32	Login/Drawer UI/LeaderBoard	12/10/15	12/16/15	Judah
33	JUnit Testing	12/10/15	12/16/15	Michaela
34	**Documentation**	**12/14/15**	**12/16/15**	
35	SRS/Final Report	12/14/15	12/16/15	Chris
36	SRS/Final Report	12/14/15	12/16/15	Jordan
37	SRS/Final Report	12/14/15	12/16/15	Judah
38	SRS/Final Report	12/14/15	12/16/15	Michaela

Figure B.3: Gantt Chart of our Progress

B.4 Team's Weekly Effort

B.4.1 Chris Taormino

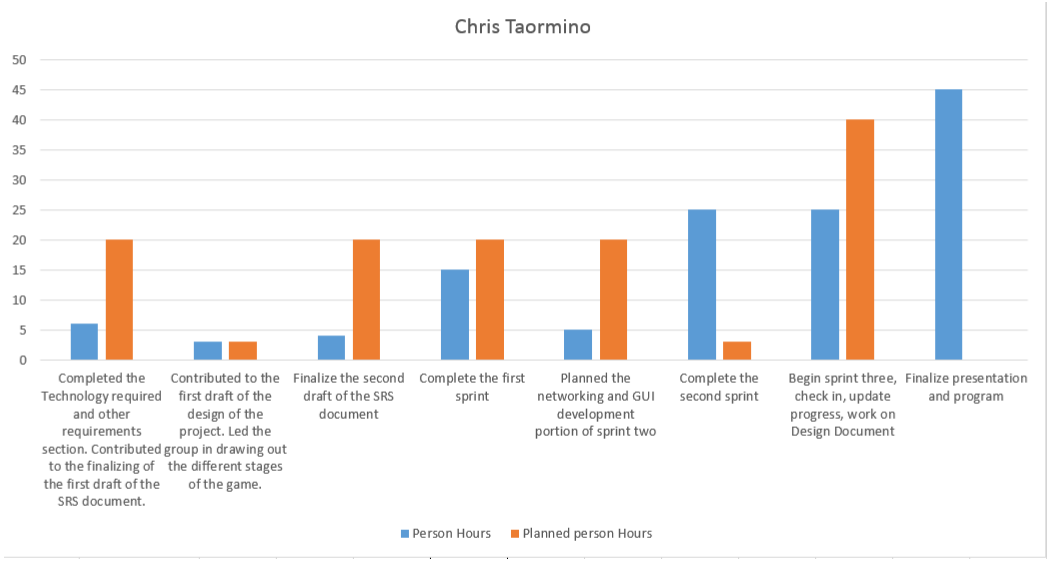


Figure B.4: Chris's Weekly Effort

B.4.2 Michaela Gates

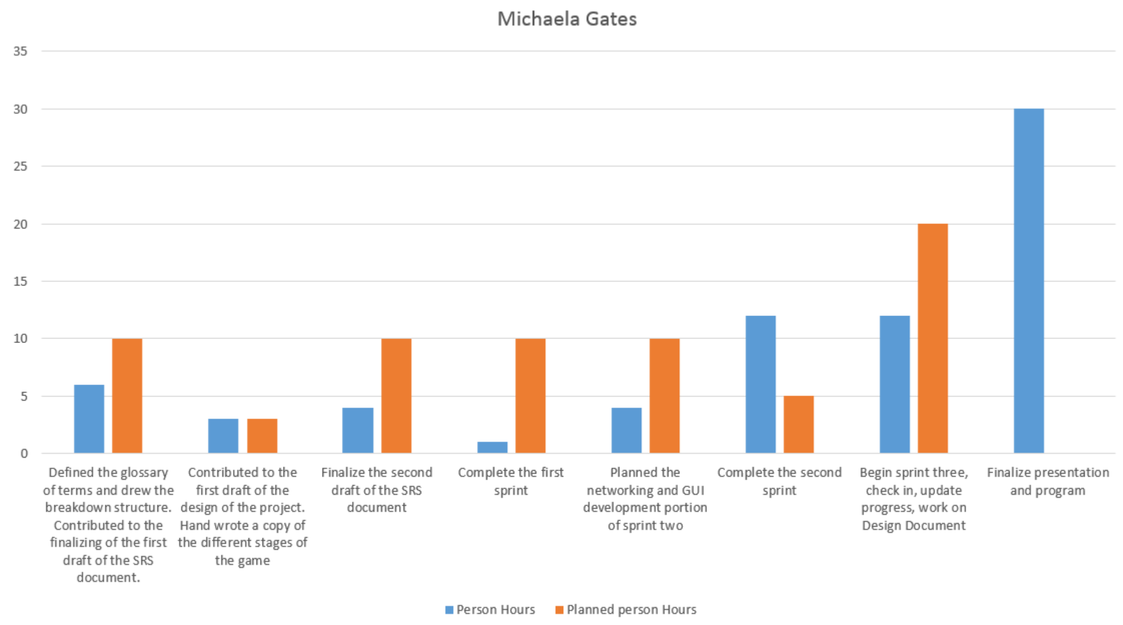


Figure B.5: Michaela's Weekly Effort

B.4.3 Judah Parham

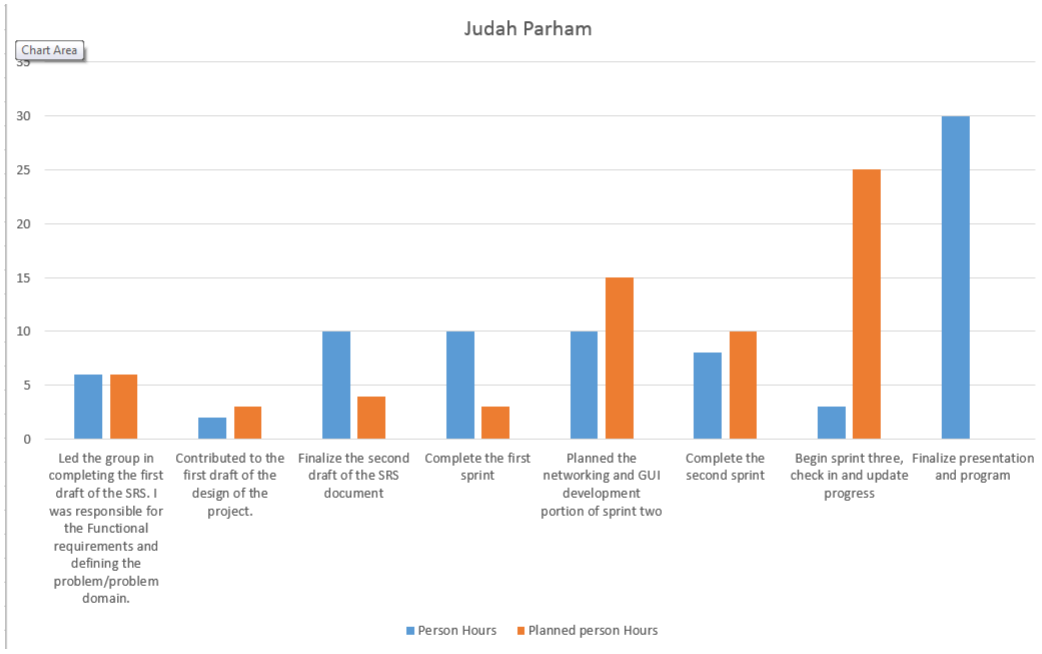


Figure B.6: Judah's Weekly Effort

B.4.4 Jordan Esty

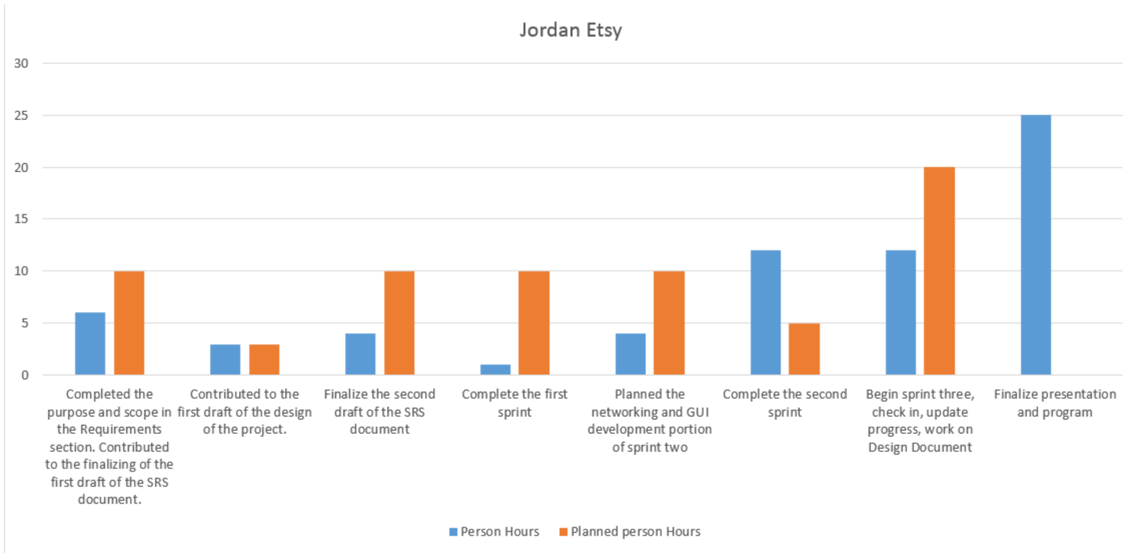


Figure B.7: Jordan's Weekly Effort

B.5 Line Diagrams

B.5.1 Chris Taormino

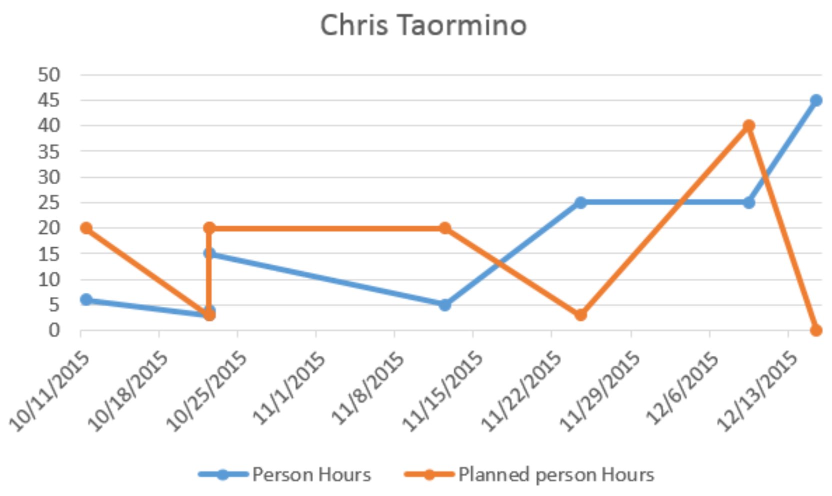


Figure B.8: Line Diagram of Chris's Man Hours

B.5.2 Michaela Gates

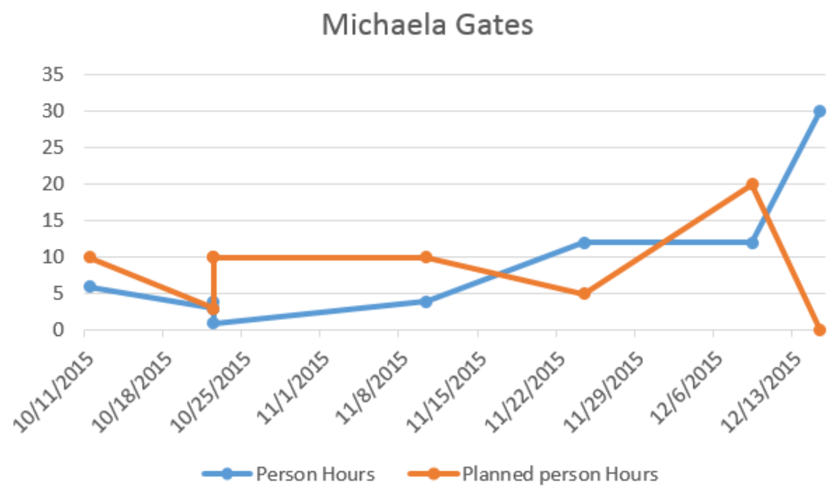


Figure B.9: Line Diagram of Michaela's Man Hours

B.5.3 Judah Parham

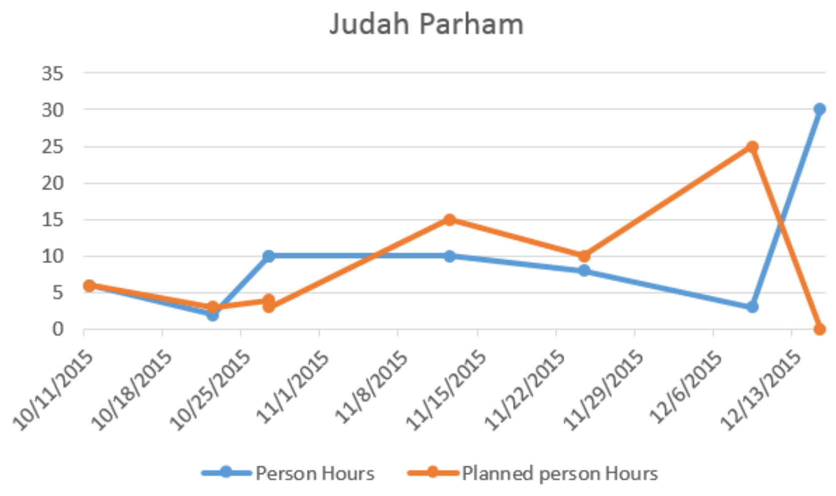


Figure B.10: Line Diagram of Judah's Man Hours

B.5.4 Jordan Esty

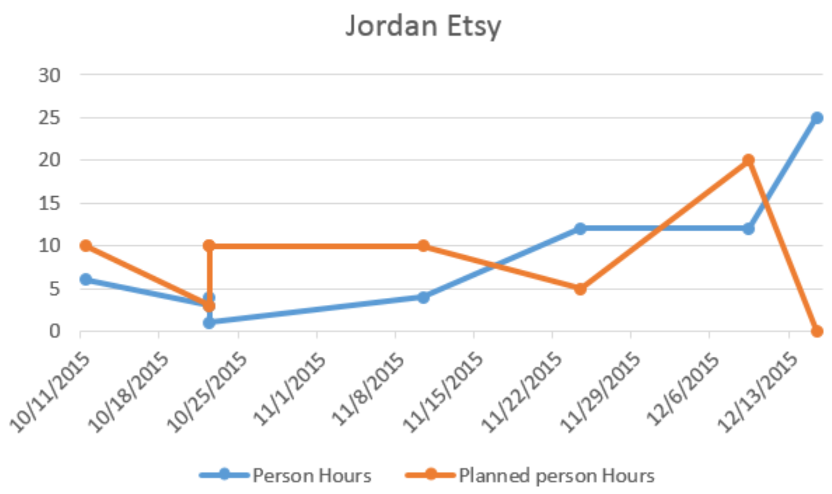


Figure B.11: Line Diagram of Jordan's Man Hours

B.6 Class Diagram

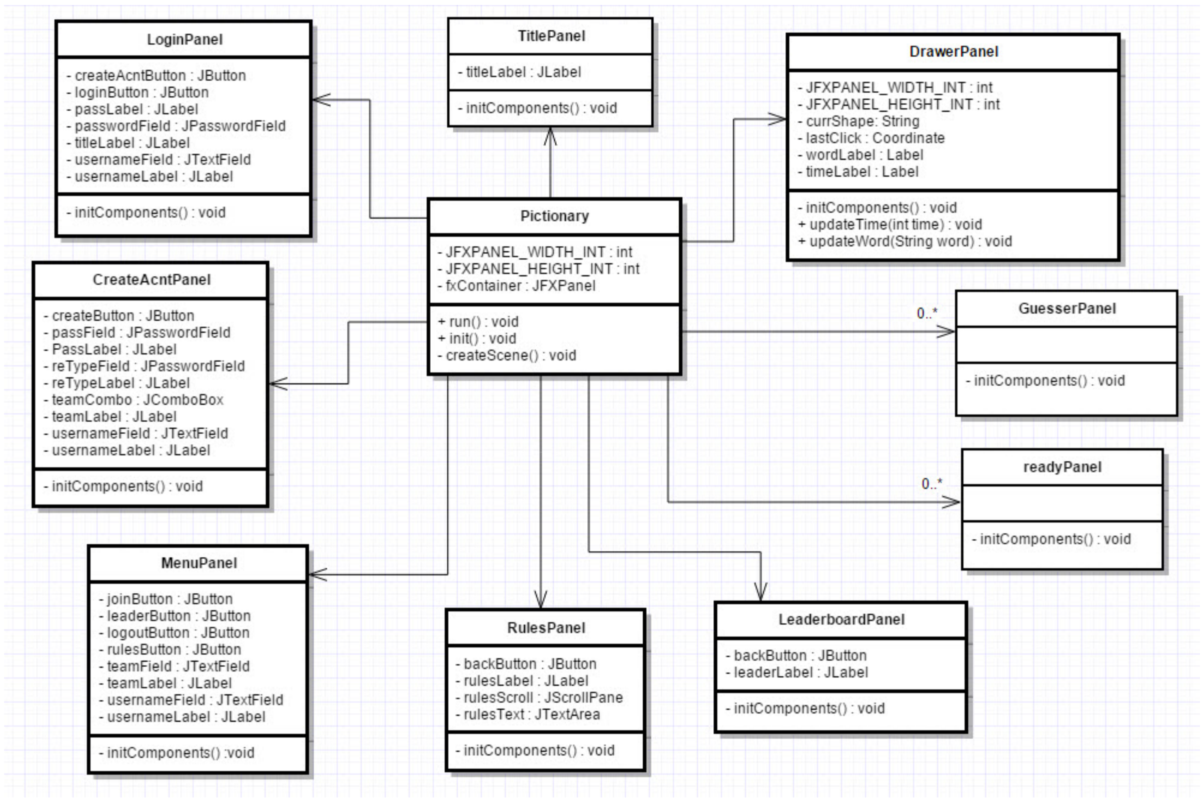


Figure B.12: UML Class Diagram of the System

B.7 Sequence Diagram

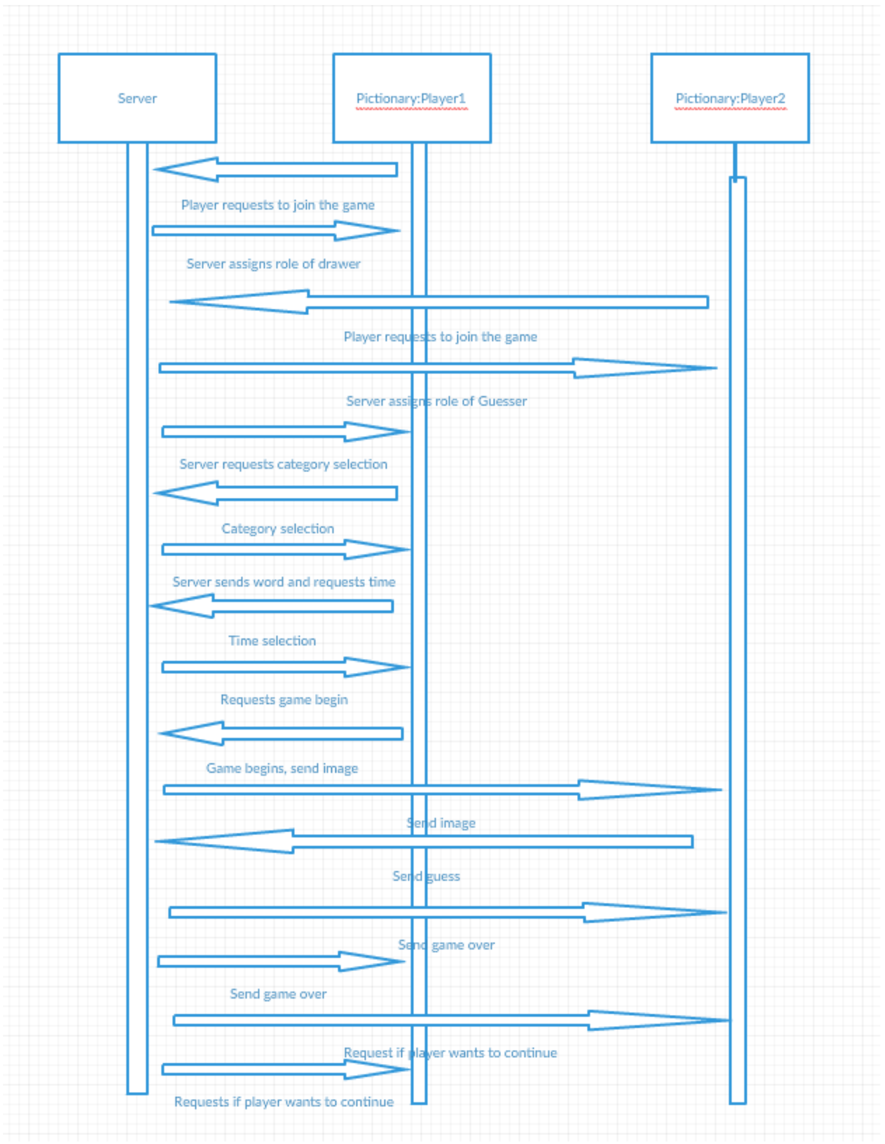

Server
Pictionary:Player1
Pictionary:Player2
Player requests to join the game
Server assigns role of drawer
Player requests to join the game
Server assigns role of Guesser
Server requests category selection
Category selection
Server sends word and requests time
Time selection
Requests game begin
Game begins, send image
Send image
Send guess
Send game over
Send game over
Request if player wants to continue
Requests if player wants to continue

Figure B.13: Sequence Diagram of Guesser guessing correctly on first try

B.8 State Diagram

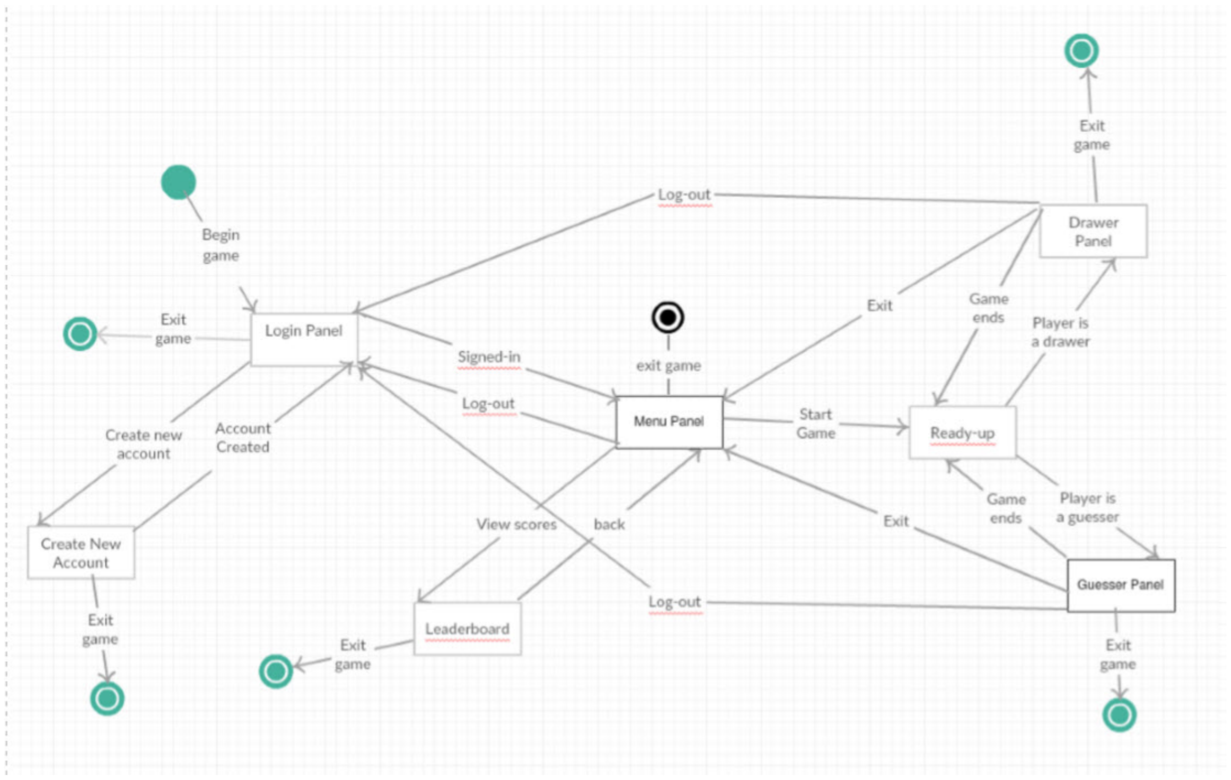


Figure B.14: State Diagram of GUI

B.9 ER Diagram

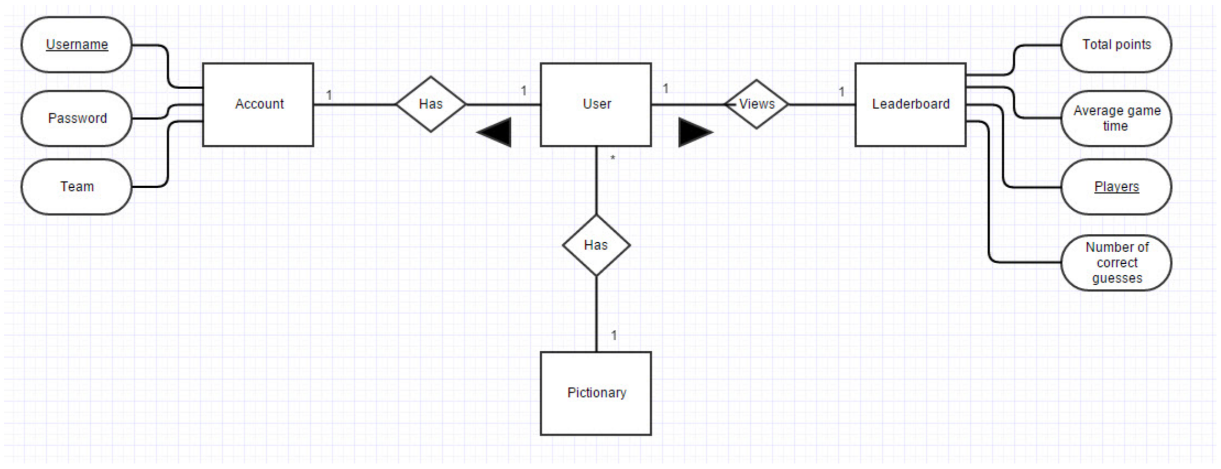


Figure B.15: ER Diagram of the System

B.10 Design Architecture

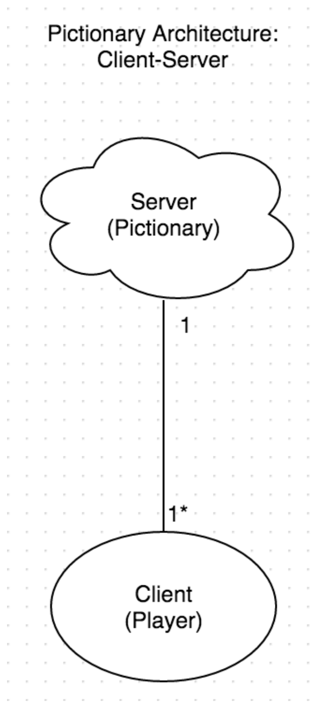


Figure B.16: Design of the Database