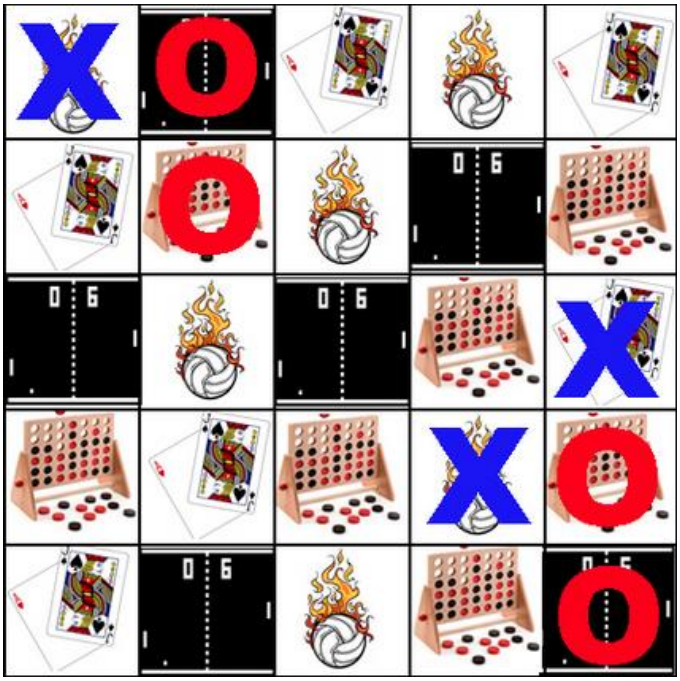
Gameception



Prepared by:
Group 4

Ed Manolache
Parth Mody
Suraj Shetty
Vishal Doshi

For CS 440
At
The University of Illinois Chicago

Fall 2013

Table of Contents

List of Figures:

I Project Description

1 Project Overview

"Gameception" is a strategized version of the classic "tic-tac-toe" game. Every square within the game consists of a mini-game, and placing an 'X' or an 'O' in any square would be determined by the outcome of the mini-game within the square. However, unlike the classic "tic-tac-toe" game the grid size is an m x m grid that can be decided by the player.

The objective of the game is similar to the objective of a regular "tic-tac-toe" game i.e to get a pre-determined number X's or O's in a row also decided by the player, either horizontally, vertically or diagonally. However, the player can place an X or an O in a square only if they win the mini-game played within the square. The mini-game itself would be a different game altogether (eg. 4-in-a-row, Pong, Blackjack, etc.).

There are various elements of strategy within the game, so that players can play the mini-games while keeping in mind their strategy to win the main "tic-tac-toe" game. Including, losing a mini-game on purpose in order to get to select the next mini-game and corresponding grid square.

The game provides two methods of gameplay, either playing online with another person or a single player mode with an Artificial Intelligence agent. The mini-games are customizable based on the user's preference and can be downloaded from the embedded game store in order to keep the players engaged through new downloads from third-party developers.

2 The Purpose of the Project

2a The User Business or Background of the Project Effort

The motivation is to develop a game of a traditional table-top format based on the study made by the Marketing and Planning Division of Mélange Computing Services. The study indicates great opportunities for the development of such games, as opposed to first person shooter games which are already saturated in the market.

2b Goals of the Project

We would like to create a product that can be successfully marketed and popularized within the current tablet and smartphone app-market.

We would also want to encourage developers to create the games for the game market from which the minigames are imported.

2c Measurement

We can measure the aforementioned goals by the number of downloads and ratings received by the game from customers.

We can also measure the popularity based on the number of developers who choose to create minigames for the product

3 The Scope of the Work

3a The Current Situation

Currently, there are 2 types of businesses in the table-top gaming world, i.e. Tablet games like the ones on iPad /Android devices or traditional table top games (board games) like checkers, chess or carom. The way business was conducted for these games was that there was one single game developed by the developers and released on the market where the market could be a toy store(for board games) or a digital marketplace(e.g. Apple's App Store and Android's Play Store). On the other hand ours will be a more continuous process where we build the basic game and a few mini-games and encourage developers to build games for our Game-Store.

3b The Context of the Work

We need to investigate the current status of the gaming industry. Basically, we need to filter out which games are popular depending on metrics and analytics ranging from ratings to downloads. Also, we may need to consider demographics while analyzing game popularity. As well as finding out strategies to attract developers to build games for us. This investigation would be a two-part process. One of them would be to analyze the current trends with developers, such as the level of comfort with certain platforms or languages. Also, analysis and investigation regarding building robust, easy to use and well documented SDK's along with an online support community needs to be done.

3c Work Partitioning

List of Business Events:

Event Name	Input and Output	Summary
Track user growth	Users tracked(I) Low growth(O)	Track user growth and if its slow, certain promotional strategies may be applied like e-mail and social media.
Track user activity	Activity tracked(I) Low activity(O)	Track user activity, and if its low, certain strategies like a weekly newsletter or a blog.
Track server health	Server health tracked(I) Current servers unable to handle load(O)	Track server health, and if more are needed, install them
Update SDK	SDK updated(I)	The SDK should be regularly updated to provide more functionality for the developers
Track 3rd party developer interest	Developer interest tracked(I) Developer interest low(O)	Our product relies heavily on 3rd party game developers. We have to keep them in the loop by organizing certain events and give them incentives.

3d Competing Products

The two primary competing entities would be board games and tablet games. As discussed before, both of these games are perfect in their own right but our product introduces a unique concept of a “game within a game” which adds a strategic element to the fun “mini-games” that we are so used to playing. Hence it completely redefines the “casual gaming experience”.

4 The Scope of the Product

4a Product Boundary

Refer III.Statement Model>Use case Model. Page Number 24.

Actors:

Name	Description
Player	The user who plays the game.
Game Center	The device's (iOS or Android) embedded game center which handles friend lists, friend invitations, leaderboards, and achievements.
Third Party Developer	Develops additional games that can be downloaded from the Game Store by the player to be used within one of the squares in the tic-tac-toe grid of games.
AI	Artificial Intelligence that can play against the user in the game.
System Administrator	Responsible for general system maintenance including the release of new updates, reviewing third party games before their acceptance in the Game Store, and solving general issues.

4b Product Use Case List

Refer III.Statement Model>Use case Model. Page Number 24.

5 Stakeholders

5a The Client

Users of smartphone / tablets	Who are interested in playing games are the customers. They can install our game-application from an online App-Store.
Third Party Developers	Who are interested in developing games for our Game-Store which in-turn would fetch them revenue.

5b The Customer

Users of smartphone / tablets	Who are interested in playing games are the customers. They can install our game-application from an online App-Store.

5c Hands-On Users of the Product

User	All the Smart-device owners interested in playing single-player or multiplayer games. Age group may range from 6 yrs. and up.
User role	Buy game from the App-Store.
Subject matter experience	Totally dependent on the user and the skill he/she has.
Technological experience	User's technological-experience may range from novice to master.

5d Priorities Assigned to Users

Key users	The users who buy our application for playing the game.
Secondary users	Developers who are interested to develop games for our embedded 'Game Store'.

5e User Participation

Gamers	Being able to play various games in just one 'Big-Game' would guarantee a good audience participation. And the scope for the developers to create new games for our game-store is very-broad. This would keep the audience attracted toward our game, Gameception.
Third-Party Developers	Being able to develop a game for our proprietary game-store, developer would have key interest in our application.

5f Maintenance Users and Service Technicians

System Administrator	Administrator: An IT Professional who would be responsible for maintaining the server.
Developers	A team of software developers who work hard to update the product regularly and provide support to third party developers.

5g Other Stakeholders

Third-Party Developers	Teams/ Individual responsible for creation and maintenance of the games they have put up on our Game Store.
Advertisers	Individual/Party interested in advertising their product/service within our game.

6 Mandated Constraints

6a Solution Constraints

Constraint 1:

Description	It would be a traditional tabletop game.
Rationale	An extensive study indicates great opportunities for our company to develop games of a traditional "tabletop" format which require strategic thinking to play and win.
Fit criterion	Large number of users.

Constraint 2:

Description	The SDK for third-party developers shall be well documented by development team.
Rationale	The Third-Party Developer will need to refer the documentation while designing games for the Game-Store.
Fit criterion	The integration process between third-party developers and our product should be seamless.

Constraint 3:

Description	The product should include the option to play against computer opponent with at least some degree of artificial intelligence.
Rationale	Marketing has determined that the game will be much more marketable in the sense that both single-player and multiplayer markets are explored.
Fit criterion	The game should be a success with both single and multiplayer gamers.

6b Implementation Environment of the Current System

Description	The product should available on application stores like Apple iTunes and Google play.
Rationale	This would give much more exposure to our game.
Fit criterion	The game should be a multi-platform success.

6c Partner or Collaborative Applications

Description	The product should have a support for Third-Party games
Rationale	The product inherently relies on third-party developers to make the experience more customizable and fun
Fit criterion	The Game Store should be a sizeable success

7 Naming Conventions and Definitions

7a Definitions of Key Terms

- Device: The users iOS or Android phone/ tablet.
- Game-Store: An embedded store within the game that will be used to download new games to play inside the tic-tac-toe grid. This is independent from the native app store associated with the user's device.
- Game Center: An online multiplayer social gaming network that allows users to invite friends to play a game, start a multiplayer game through matchmaking, track their achievements, and compare their high scores on a leader board. It is implemented by the device itself (e.g. iPhone, iPad, etc).
- Grid: The main frame consisting of a m X m matrix of squares in which the game is played.
- Square: Individual sections of the grid which contain embedded games.
- Mini-game: the embedded games within the squares.
- Symbol: A symbol is either an 'X' or an 'O' on the grid.

8 Relevant Facts and Assumptions

8a Facts

- Within the Game Store, Developers will be able to create new games and market them towards players in which they would be able to purchase these games for an additional fee if imposed by the developers.
- Third Party Developers will be provided with an SDK for creating new games and releasing them to the Game Store.

8b Assumptions

- Players will not need to create an account within the game itself because those services will be provided based on their device's native Game Center, such as using the iOS Game Center. Assuming that Google integrates their Google Play Services by the time the game is released.
- Players will have the option to use their device's native Game Center matchmaking services to play with their friends, to be randomly matched with another player, or play with an AI.
- Players will be able to purchase additional games within the embedded Game Store using their devices native marketplace which allows in-app purchases.

II Requirements

9 Functional Requirements

9.1. The application should allow user to sign-in via Game Center
9.2. The application should include the following four mini-games by default: 4-in-a-row, pong, blackjack, and tic-tac-toe.
9.3. The application should allow the user to play in either single player (against an AI agent) or multiplayer mode.
9.4. The application should allow the user to select the size of the grid, with the minimum grid size being 3 x 3.
9.5. The application should allow the user to select the number of symbols in a row required to win a game, the minimum being 3 and the maximum being the number of grid rows the user selected.
9.6. The player should be able to choose which mini games can be played.
9.7. The player should be able to download third-party games from the native Game-Store.
9.8. The player should receive notifications when another player wants to play with them through the Game-Center.
9.9. The player should be able to view his own statistics as well as view a leaderboard with the top players.
9.10. The user should be able to access settings menu to mute the sound.

10 Performance Requirements

10a Speed and Latency Requirements

10.a.1. The application should boot up within 10 seconds
10.a.2. The application should be able to establish a connection between two opponents in a multiplayer game within 10 seconds
10.a.3. The application should be able to function with a minimum supplied bandwidth of 2 mbps over Wi-Fi or cellular data.

10b Precision or Accuracy Requirements

10.b.1. The application should allow the user to accurately select the grid square she wishes to choose.

10c Capacity Requirements

10.c.1. Initially, the application should be able to support a 1000 players. The capacity can be increased later based on the increase in traffic.

11 Dependability Requirements

11a Reliability Requirements

11.a.1. The application must be have reliable and consistent performance.
11.a.2. The application should be monitored by software engineers and make sure it complies with customer's expectations.

11b Availability Requirements

11.b.1. The application should be available for use at all times.

11.b.2. The application should have uptime of 98%; the downtime maybe due to application updates or fixes.

11c Robustness or Fault-Tolerance Requirements

11.c.1. In case of no network connectivity, the application should operate in single player mode.

11.c.2. In case of slow network connection, the application should still perform effectively.

11.c.3. The application should be fault tolerant and in case of failure, it should restore back to the last running state.

11d Safety-Critical Requirements

11.d.1. The application should not cause light-induced seizure.

11.d.2. The mini-game should be thoroughly verified for absence of any malicious code.

12 Maintainability and Supportability Requirements

12a Maintenance Requirements

12.a.1. The application should be able to accept updates via the App Store hence fixing any bugs or issues in the previous release.

12b Supportability Requirements

12.b.1. A help section describing various concepts and interactions in the game should be a part of the application.

12.b.2. There should be a feedback section in the app which allows the users to submit their feedback about flaws in the game.

12.b.3. There should also be an online portal for customer support.

12.b.4. The application should be compatible with both phones and tablets.

12c Adaptability Requirements

12.c.1. The application should be able to run on both Android and iOS devices.

12.c.2. The application should be able to work both on Wi-Fi or cellular data.

12d Scalability or Extensibility Requirements

12.d.1. The application should be robust enough to handle any given load of users.

12e Longevity Requirements

12.e.1. The product shall be expected to operate with proper maintenance, and within the preset budget for a minimum of 5 years.

12.e.2. Frequent updates to the game would ensure stability and responsiveness in the game.

13 Security Requirements

13a Access Requirements

13.a.1. Player information; only including the player's username should be accessed by the application.

13.a.2. Players should be able to access their own statistics including; leaderboard position, win-loss ratio, etc.

13.a.3. Players should only be allowed to access other player's username, and game statistics including leaderboard position and win-loss ratio through the Game Center.

13.a.4. The application should require authorization in order to make mini game purchases through the application's Game Store using the devices in app purchasing paradigm, which should be restricted for those users who have not allowed in app purchasing on their device.

13.a.5. The application should not be able to access the user's credit card information when making purchases through the in app purchases, since the user's device will provide a framework to notify the application when a mini game has been purchased.

13b Integrity Requirements

13.b.1. The server should make daily backups of all the third party mini games that are present in the Game Store to prevent data loss.

13.b.2. The server should make daily backups of all player statistical information to prevent data loss.

13.b.3. The application should not be susceptible to any form of external attacks.

13.b.4. In a multiplayer game if one of the players quits the application or is disconnected from the internet the application should allow the player two minutes to reconnect before the game is forfeited.

13c Privacy Requirements

13.c.1. The application should not divulge any user information to anyone including any third-party developers.

13.c.2. The application should inform the users of all the privacy policies when the user first launches the application through the the presentation of the End User License Agreement.

13.c.3. The application should inform the user when the privacy policy has changed and show the user the modifications.

13d Immunity Requirements

13.d.1. The application's Security Team should review each third-party mini game before it is accepted to the application's Game Store thoroughly to ensure there is no malicious software embedded into the mini games, such as viruses, worms, Trojan horses, and adware.

14 Usability and Humanity Requirements

14a Ease of Use Requirements

14.a.1. The user should be able to download the application easily from the iTunes Store or Android Play Store.

14.a.2. The application should be easy to use for people of all ages from 6 years and above

14.a.3. The application can be used by anyone who uses a smartphone or tablet device

14.a.4. The application design would be intuitive to allow the user to use it with little to no help or remembrance from the previous use

14.a.5. The application should provide an immediate response for the action the user has performed to assure the user that it is functioning according to her actions.

14.a.6. The application should intimate the user if she commits an error while using the device and provide feedback on how to revert the error.

14b Personalization and Internationalization Requirements

14.b.1. The player should be able to select the grid size depending on her preferences.

14.b.2. The player should be able to select the number of symbols on arrow required to win.

14.b.3. The player should be allowed select a few Mini Games that would be a part of the grid in the main game.

14c Learning Requirements

14.c.1. As mentioned earlier in the ease of use requirements, any user who has experience in using modern touch based smart devices can use the application without any learning curve whatsoever.

14.c.2. Prior smart device users should find it easy to learn to use the application.

14.c.3. Also for users who aren't prior smart device users should be able to use it within a short span of time as the application is intuitive inherently.

14d Understandability and Politeness Requirements

14.d.1. The application should use symbols and words that are understandable by the user

14.d.2. The application should provide an interface that would offer informative feedback in a polite context

14.d.3. The application should provide positive reinforcements for correct actions, if the user is using the device for the first time

14.d.4. The application should not reveal any internal details of its construction and code to the user, and should not be intimidating to use.

14e Accessibility Requirements

14.e.1. The application should be easy to use for people with common disabilities such as color-blindness, myopia and partially sighted users.

14.e.2. The application should be easy to use for elderly people by using larger texts and icons where possible, which could be easily readable.

14f User Documentation Requirements

14.f.1. The application support website should provide the user with installation instructions to install the application from the Play Store or iTunes Store

14.f.2. The Help screen should provide general information for the user to use the game and how to download games from the Game-store.

14.f.3. Each Third-Party developer should provide a quick help screen for that particular Mini-Game.

15 Interface Requirements

15.1. The Third Party developer should be provided precise set of rules and instructions for development of acceptable mini-game.

15.2. The mini-game submitted by Third-Party developers should be reviewed thoroughly before making it available to application users.

16 Look and Feel Requirements

16a Appearance Requirements

16.a.1. The color scheme used must neither be too bright, nor too dull.

16.a.2. The font size must neither be too small nor too large.

16.a.3. The appearance of product should rather be sophisticated than naive.

16b Style Requirements

16.b.1. The color scheme used must neither be too bright, nor too dull.

16.b.2. The font size must neither be too small nor too large.

16.b.3. The appearance of product should rather be sophisticated than naive.

17 Operational and Environmental Requirements

17a Expected Physical Environment

17.a.1. The application can be used in any given physical environment with sufficient access to the internet.

17b Requirements for Interfacing with Adjacent Systems

17.b.1. The application should work in tandem with the GameCenter portal in iOS and its equivalents in Android in future scope.

17.b.2. The application should make use of various services provided by GameCenter like push notifications, social networking services etc.

17c Productization Requirements

17.c.1. The application should be distributed like any other app on the App Store area.

17d Release Requirements

17.d.1. Updates or releases will not be time based. Rather they will be for the purpose of adding new features and removing bugs.

18 Legal Requirements

18a Compliance Requirements

18.a.1. The Instances product will abide by the laws in all aspects.

18.a.2. The will adhere with all the legal requirements.

18b Standards Requirements

18.b.1. The product will adhere to all the Apple iOS and Android mobile standards.

18.b.2. The product will be Apple's Game Center - Aware game.

19 Scenarios

Scenario name	selectingTheMiniGames
Participating Actor	Alice: Player
Flow of Events	1. Alice opens the Gameception Application. 2. She selects the settings button and is able to see the Settings page. 3. Under Mini-Games, she is shown the list of mini-games that are available to her. 4. She selects the games that she would want to be a part of the main grid.

Figure 1*: selectingTheMiniGames scenario for Gameception*

Scenario name	startingASinglePlayerGame
Participating Actor	Alice: Player, Bot1: AI
Flow of Events	1. Alice clicks on the Single Player game option. 2. Alice chooses the grid size to be 4 x 4.She then selects the number of symbols in a row required to win a game to be 3. 3. The game generates the mini games on the grid and selects a mini game for each individual square on the grid. Half of the games selected are based on Alice's settings regarding which games she prefers to play and the other half are randomly selected. 4. The game randomly selects Bot1 to make the first move.

Figure 2*: startingASinglePlayerGame scenario for Gameception*

Scenario name	startAutoMatchGame
Participating Actor	Alice: Player, Chris: Player
Flow of Events	1. Alice selects the multiplayer game option . 2. Alice chooses the grid size to be 5x 5. 3. She then selects the number of symbols in a row required to win a game to be 4. 4. She is presented with 2 options "Invite Friend" and "Auto Match" 5. She selects the "Auto Match" option. 6. The Game Center then finds Chris who is not Alice's friend as an appropriate match based on picking the same grid size, picking the same symbols in a row required to win a game, and number of common games. 7. Both players are presented a 5 x 5 grid, wherein a third of the squares in the grid are games selected based on Alice's choices, another third of the squares in the grid are selected based on Chris' choices and the rest are randomly picked. 8. The game randomly selects Alice to make the first move.

Figure 3*: startAutoMatchGame scenario for Gameception*

Scenario name	startGameWithAFriend
Participating Actor	Alice:Player, Bob:Player
Flow of Events	1. Alice selects the multiplayer game option. 2. Alice chooses the grid size to be 4 x 4. 3. She then selects the number of symbols in a row required to win a game to be 3. 4. She is presented with 2 options "Invite Friend" and "Auto Match" 5. She selects the "Invite Friend" option and is shown the Game Center interface to select a friend. 6. After she selects the friend "Bob" , the friend is sent an invitation. 7. The friend accepts the invitation. 8. Both players are presented a 4 x 4 grid,wherein a third of the squares in the grid are games selected based on Bob's choices,another third of the squares in the grid are selected based on Alice's choices and the rest are randomly picked all of which can be previewed before picking them. 9. The game randomly selects Bob to make the first move.

Figure 4*:startGameWithAFriend scenario for Gameception*

Scenario name	PlayingMiniGame/Win
Participating Actor	Alice:Player, Bob:Player
Flow of Events	1. Alice starts a 4x4 game with a 3-in-a-row win rule with Bob. 2. She notices that the game on square (1,1) is Pong and selects that to be played between her and Bob 3. In Pong, each player has a paddle that moves vertically on two opposite edges of the playing surface with a ball moving from one edge to the other. 4. The players move their paddles to prevent the ball from going beyond their edge, failing which the opposing player gets a point. 5. The winner in the game would be the first to score 11 points. 6. Alice scores 11 points first and wins the minigame. 7. This places Alice's symbol 'X' on the square at location (1,1) 8. Bob now gets to select the next square to play in.

Figure 5*: PlayingMiniGame/Win scenario for Gameception*

Scenario name	PlayingMiniGame/Lose
Participating Actor	Alice:Player, Bob:Player
Flow of Events	1. Alice starts a 4x4 game with a 3-in-a-row win rule with Bob. 2. Alice having won the previous minigame at (1,1), it is Bob's turn to select the square to be played in. 3. Bob notices the minigame in square (1,2) is tic-tac-toe, a game he believes he is good at, and selects the square in order to block a horizontal 3-in-a-row in the main game for Alice. 4. Alice manages to beat Bob in the mini-game as well by getting 3-in-a-rowThis places Alice's symbol 'X' in the square at location (1,2) which, paired with her previous 'X' at (1,1) gets her 2 X's in a row. 5. Since Alice won the minigame, it is again Bob's turn to select the next square to play.

Figure 6*: PlayingMiniGame/Lose scenario for Gameception*

Scenario name	DownloadingANewMiniGame
Participating Actor	Alice:Player
Flow of Events	1. Alice selects the Game Store option in the app. 2. She is presented with a catalog of games from which to select. 3. She selects game "Slime Volleyball" to download and is asked to enter her Apple Id password or Google password(Wallet) for her payment information. 4. After her credentials are authenticated the game is added to her profile. 5. By default the game is selected as a possible minigame in the grid which can be deselected by her in the settings.

Figure 7*: DownloadingANewMiniGame scenario for Gameception*

Scenario name	PlayingTheMainGame/Win
Participating Actor	Alice:Player, Chris: Player
Flow of Events	1. Alice is playing a 5x5 game with a 4-in-a-row win with Chris. 2. Alice has won grids (1,1),(1,2),(1,3),(4,5) and (3,4). 3. She notices that winning the mini-game at (1,4) would make her winner of the "Main-Game" as she will be getting 4-in-a-row. 4. So with a strategy to win the "Main-Game", she deliberately loses a mini-game at (4,4) and now she gets to select the next location where the game will be played. 5. Alice selects (1,4) as the next location for mini-game. 6. Alice wins the minigame. 7. Now, having 4-in-a-row i.e. (1,1),(1,2),(1,3) and (1,4), Alice "Wins" the "Main-Game".

Figure 8*: PlayingTheMainGame/Win scenario for Gameception*

20 Use Case Model

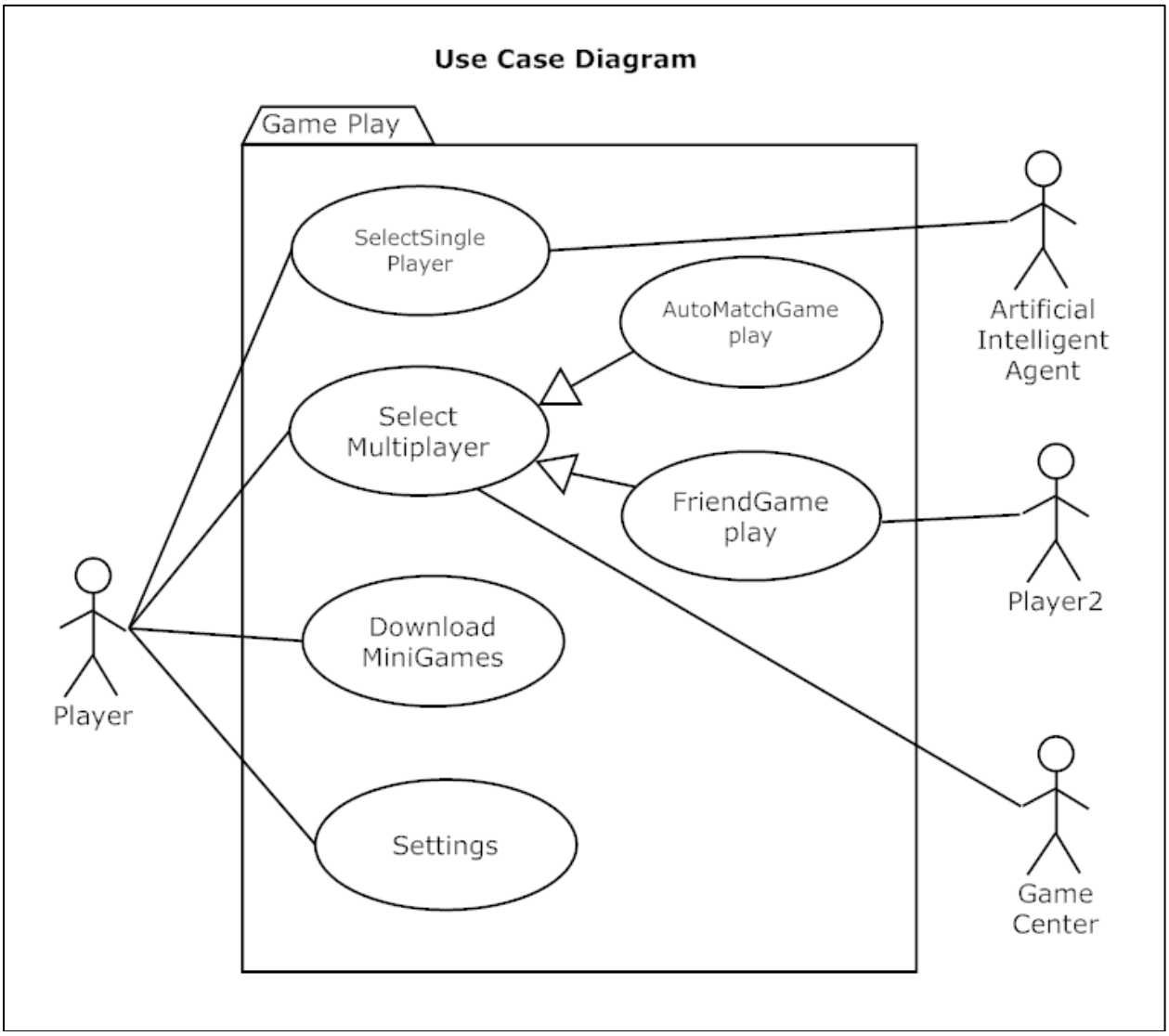


Figure 9*: Game Play Use-Case Diagram*

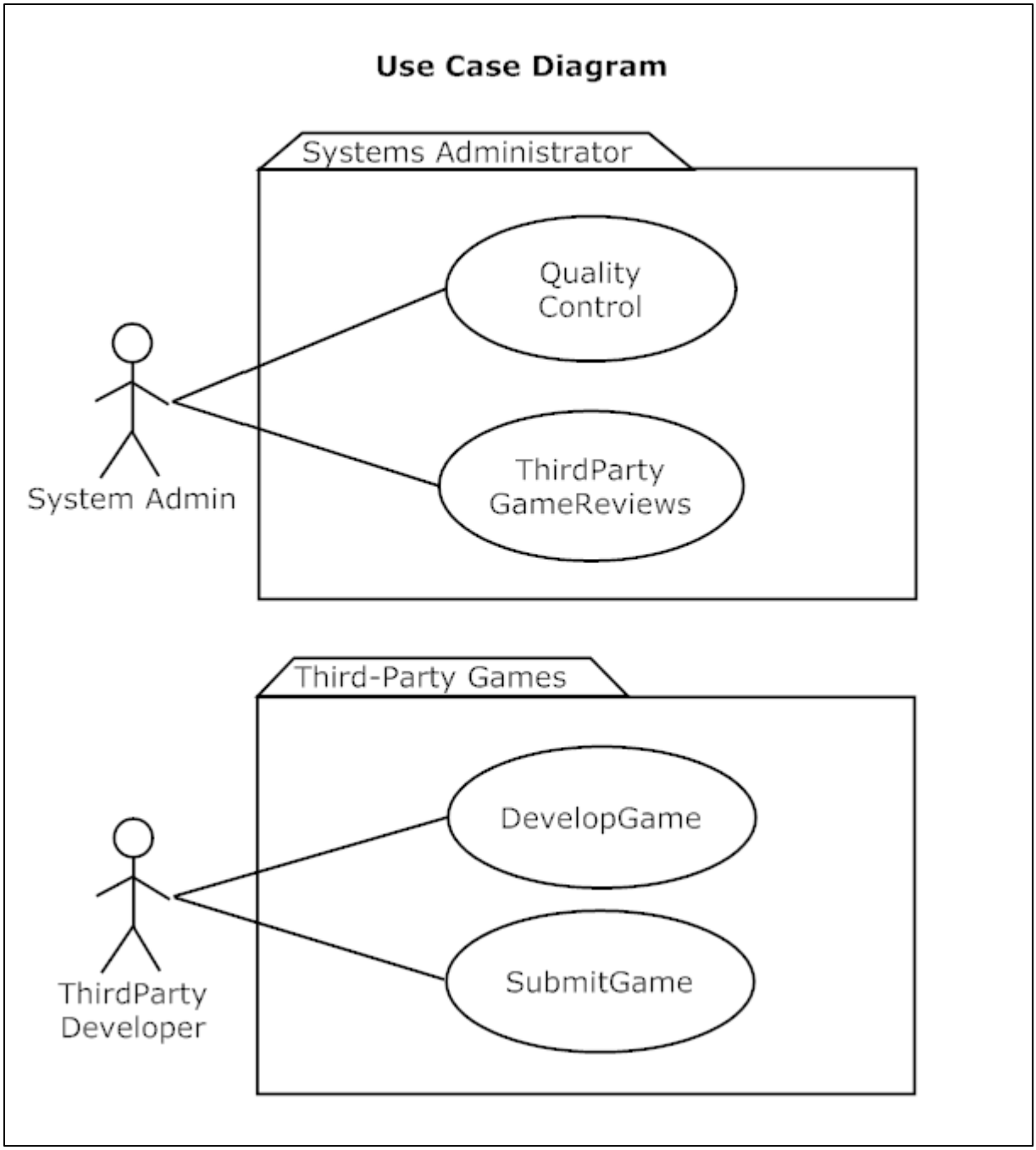


Figure 10*: System Administrator &Third-Party Games Use-Case Diagram*

Use Cases:

Use Case 1:

Use Case:	SelectSinglePlayer
Actors:	Player and AI
Entry Conditions:	The player must be signed in through his/her device's Game Center
Flow of Events:	1. The player selects the m x m size of the grid. 2. The player selects the number of symbols in a row required to win a game. 3. The mini games are randomly generated for each of the squares on the tic-tac-toe grid based half of the player's picked games based on his/her settings configuration and the other half of the games are randomly selected. 4. The game randomly selects either the user or the AI to make the first move. 5. The player with the first move gets assigned the symbol 'X' and the second player will be assigned the symbol 'O'. 6. The first move consists of either the user or the AI deciding on a mini game strategically based on either a desired square position or a game they are confident in winning. 7. The player and AI play the mini game based on the particular rules for that mini game until there is a winner. 8. The winner of the mini game assigns their symbol on that particular square. 9. The loser of the mini game selects the next mini game to be played using a strategy to try and either put their symbol on a certain square in order to try and win the tic-tac-toe game or pick a mini game they are confident in winning. 10. Steps 7 through 9 are repeated until the remainder of the squares in the grid are either completed and the game ends in a tie or there is a player who succeeds in placing the pre-decided number of respective symbols in a horizontal, vertical, or diagonal row wins the game.
Exit Conditions:	The Statistics page is updated to reflect the outcome of the game for the player.

Use Case 2:

Use Case:	SelectMultiplayer
Actors:	2 Players, Game Center
Entry Conditions:	The player must be signed in through his/her device's Game Center
Flow of Events:	1. The player has the option to play a two-player game with either their friends (in which case they select one friend from a list of friends from their device's Game Center) or an auto matched game (in which the game automatically matches two players who have picked the same grid size, the same number of symbols in a row required to win the game, and those players who have the most games in common). 2. Once two players are matched together the mini games are generated for each of the squares on the tic-tac-toe grid based on which mini games both players own. and they have selected in their individual settings configuration. One third of the games selected will be based on the games one individual has selected in their settings configuration, the next third of the games will be come from the other player's settings configuration, and the last third will be randomly selected based on which games both players have in common but are not necessarily selected in their settings configuration file. 3. The game randomly selects one of the players to make the first move. 4. The player with the first move gets assigned the symbol 'X' and the second player will be assigned the symbol 'O'. 5. The first move consists of the player deciding on a mini game strategically based on either a desired square position or a game they are confident in winning. 6. The two players play the mini game based on the particular rules for that mini game until there is a winner. 7. The winner of the mini game assigns their symbol on that particular square. 8. The loser of the mini game selects the next mini game to be played using a strategy to try and either put their symbol on a certain square to try and win the tic-tac-toe game or pick a mini game they are confident in winning. 9. Steps 6 through 8 are repeated until the remainder of the squares in the grid is either completed and the game ends in a tie or there is a player who succeeds in placing three respective symbols in a horizontal, vertical, or diagonal row wins the game.

Exit Conditions:	The Statistics page is updated to reflect the outcome of the game for each player.
Requirements:	The player must have a Game Center account set up on their device.

Use Case 3:

Use Case:	AutoMatchGameplay
Actors:	Inherited from SelectMultiplayer
Entry Conditions:	The player must be signed in through his/her device's Game Center
Flow of Events:	1. The player clicks on Multiplayer game and the device's embedded Game Center pops up. 2. The player clicks on Auto Match. 3. The player selects the m x m size of the grid. 4. The player selects the number of symbols in a row required to win a game. 5. The player is automatically matched with another player who have picked the same grid size, the same number of symbols in a row required to win the game, and those players who have the most games in common. 6. The mini games are generated for each of the squares on the tic-tac-toe grid based on which mini games both players own and they have selected in their individual settings configuration. One third of the games selected will be based on the games one individual has selected in their settings configuration, the next third of the games will be come from the other player's settings configuration, and the last third will be randomly selected based on which games both players have in common but are not necessarily selected in their settings configuration file. 7. The game randomly selects one of the players to make the first move. 8. The player with the first move gets assigned the symbol 'X' and the second player will be assigned the symbol 'O'. 9. The first move consists of the player deciding on a mini game strategically based on either a desired square position or a game they are confident in winning. 10. The two players play the mini game based on the particular rules for that mini game until there is a winner. 11. The winner of the mini game assigns their symbol on that particular square. 12. The loser of the mini game selects the next mini game to be played using a strategy to try and either put their symbol on a

	certain square to try and win the tic-tac-toe game or pick a mini game they are confident in winning. 13. Steps 10 through 12 are repeated until the remainder of the squares in the grid is either completed and the game ends in a tie or there is a player who succeeds in placing three respective symbols in a horizontal, vertical, or diagonal row wins the game.
Exit Conditions:	The Statistics page is updated to reflect the outcome of the game for each player.
Requirements:	The player must have a Game Center account set up on their device.

Use Case 4:

Use Case:	FriendGameplay
Actors:	Inherited from SelectMultiplayer
Entry Conditions:	The player must be signed in through his/her device's Game Center
Flow of Events:	1. The player clicks on Multiplayer game and the device's embedded Game Center pops up. 2. The player clicks on invite a friend. 3. Game Center shows a list of friends the player has at which point they can select a friend to invite to play with them. 4. The player selects the m x m size of the grid. 5. The player selects the number of symbols in a row required to win a game. 6. Player 2 accepts the friend invite. 7. The two friends are matched together and the mini games are generated for each of the squares on the tic-tac-toe grid based on which mini games both players own and they have selected in their individual settings configuration. One third of the games selected will be based on the games one individual has selected in their settings configuration, the next third of the games will be come from the other player's settings configuration, and the last third will be randomly selected based on which games both players have in common but are not necessarily selected in their settings configuration file. 8. The game randomly selects one of the players to make the first move.

	9. The player with the first move gets assigned the symbol 'X' and the second player will be assigned the symbol 'O'. 10. The first move consists of the player deciding on a mini game strategically based on either a desired square position or a game they are confident in winning. 11. The two players play the mini game based on the particular rules for that mini game until there is a winner. 12. The winner of the mini game assigns their symbol on that particular square. 13. The loser of the mini game selects the next mini game to be played using a strategy to try and either put their symbol on a certain square to try and win the tic-tac-toe game or pick a mini game they are confident in winning. 14. Steps 11 through 13 are repeated until the remainder of the squares in the grid is either completed and the game ends in a tie or there is a player who succeeds in placing three respective symbols in a horizontal, vertical, or diagonal row wins the game.
Exit Conditions:	The Statistics page is updated to reflect the outcome of the game for each player.
Requirements:	The player must have a Game Center account set up on their device.

Use Case 5:

Use Case:	DownloadMiniGames
Actors:	Player
Flow of Events:	1. The player is presented a list of various mini game categories that they can choose from (e.g. Board, Card, Puzzle, Sports, etc). 2. The player selects a category and can browse through those games within the category. 3. The player selects one of the mini games to purchase using their device's in app purchasing.
Exit Conditions:	The player is charged for the game and the game is then downloaded to their device and is selectable in their individual settings configuration.
Requirements:	The user has enabled in-app purchases within their devices settings.

Use Case 5:

Use Case:	Settings
Actors:	Player
Entry Conditions	User wants to change settings
Flow of Events:	1. In game sounds can be turned on/off. 2. The player has a list of mini games that they own and have downloaded to their device with check marks next to each particular game which lets the player select/deselect a game from appearing as one of the mini games within a particular square in the tic-tac-toe grid.
Exit Conditions:	The settings the player changes are saved.

Use Case 6:

Use Case:	DevelopGames
Actors:	Third Party Developers
Entry Conditions:	The Third Party Developers download the provided SDK.
Flow of Events:	1. The Third Party Developers use the SDK provided to create mini games based on their own ideas while meeting required specifications.
Exit Conditions:	The Third Party Developers submit their mini game to be reviewed.

Use Case 7:

Use Case:	QualityControl
Actors:	System Administrator
Entry Conditions:	There are support tickets that describe bugs within the application.
Flow of Events:	1. The System Administrator reviews support tickets. 2. System Administrator fixes any linger bugs based on the support tickets within the application.
Exit Conditions:	A patch is sent out in order for users to update their application and fix any bugs found by the System Administrator.

Use Case 8:

Use Case:	ThirdPartyGameReviews
Actors:	System Administrator
Entry Conditions:	A Third Party Developer has created and submitted a mini game to be reviewed.
Flow of Events:	1. The System Administrator downloads the mini game. 2. The System Administrator tests the game thoroughly to ensure it meets the necessary requirements, including that the mini provides two player multiplayer, and AI Single Player, it contains no malware, and it is a fun game that users will enjoy playing.
Exit Conditions:	The System Administrator either approves the mini game and it is uploaded to the applications Game Store or it is denied and the Third Party Developer is notified.

21 Class Diagrams

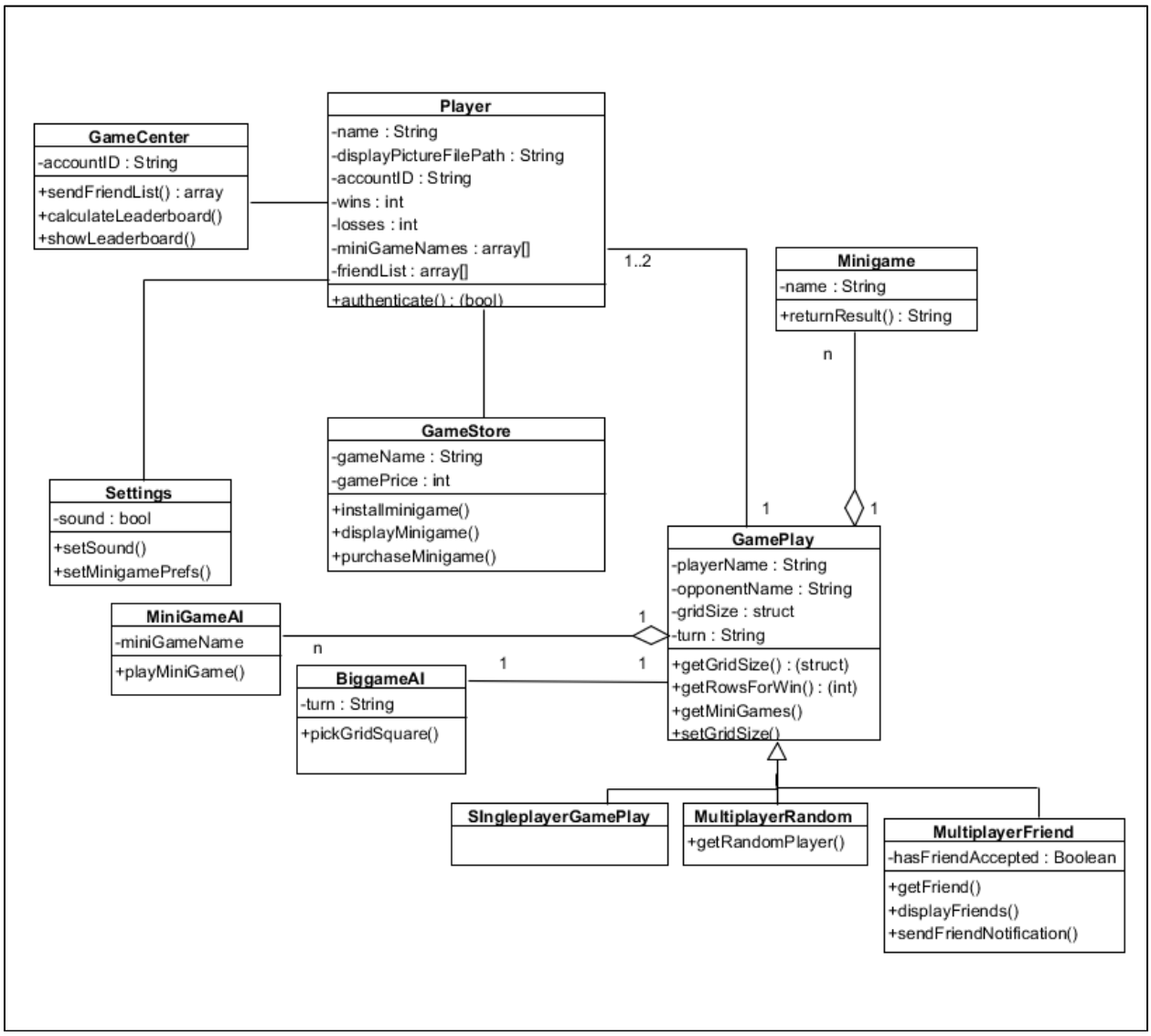


Figure 11*: Class Diagram for Gameception*

22 Dynamic Model

State Diagrams

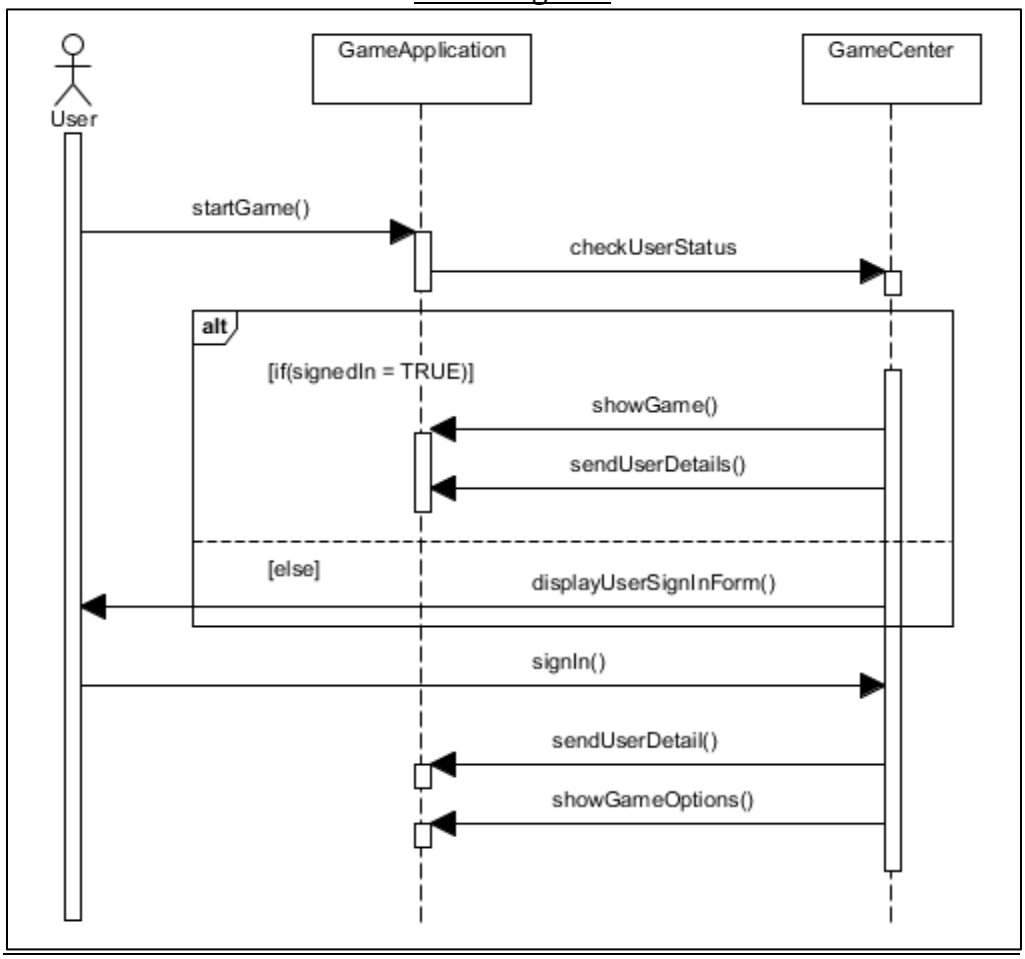


Figure 12*: Sign In Sequence Diagram*

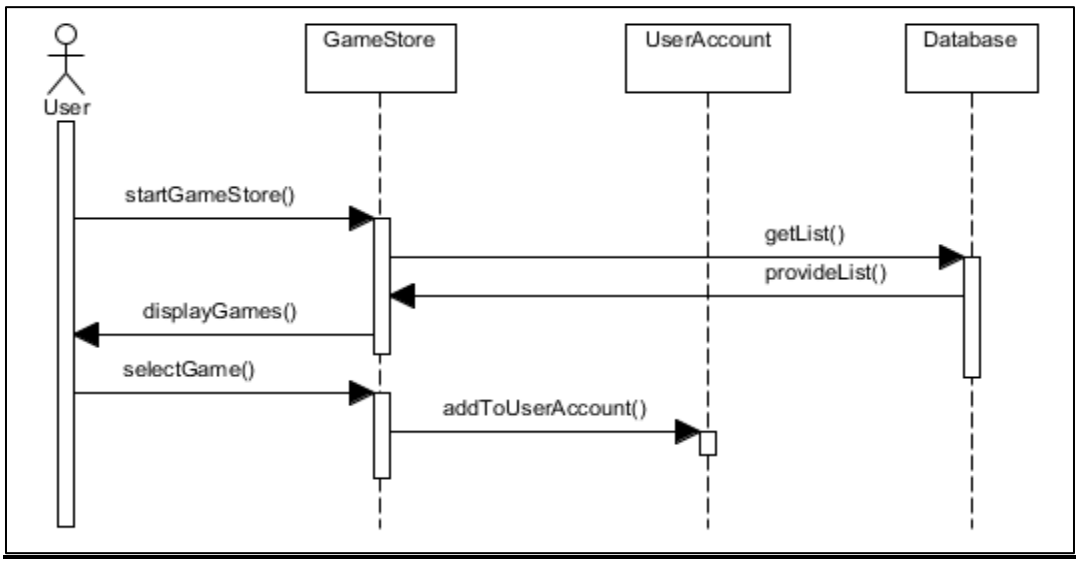


Figure 13*: Downloading Mini-Game Sequence Diagram*

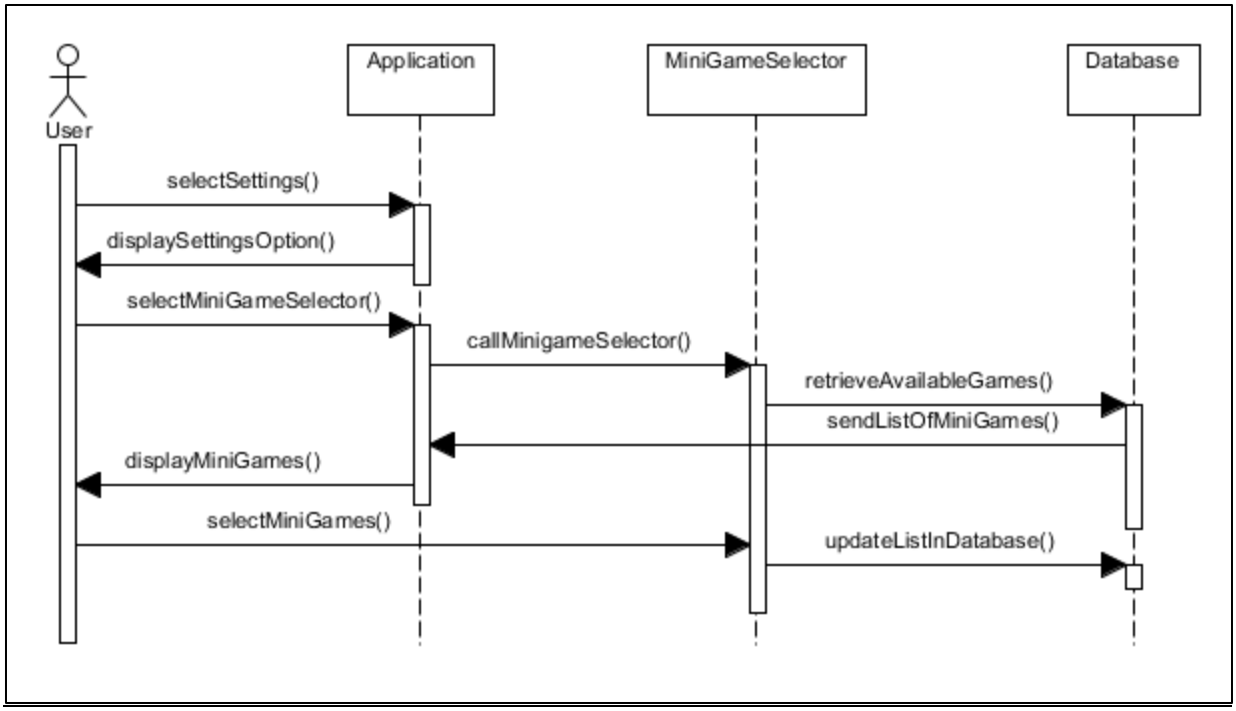


Figure 14*: Select Mini-Game Sequence Diagram*

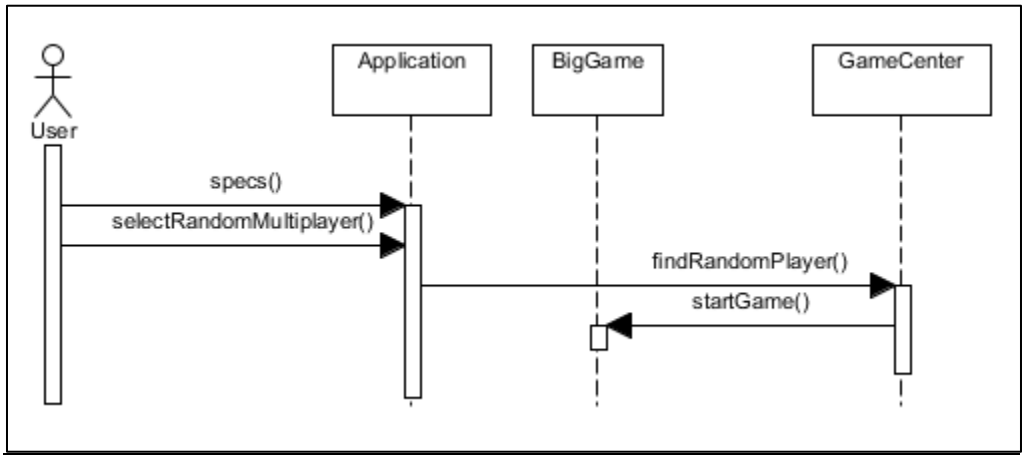


Figure 15*: Auto Match Sequence Diagram*

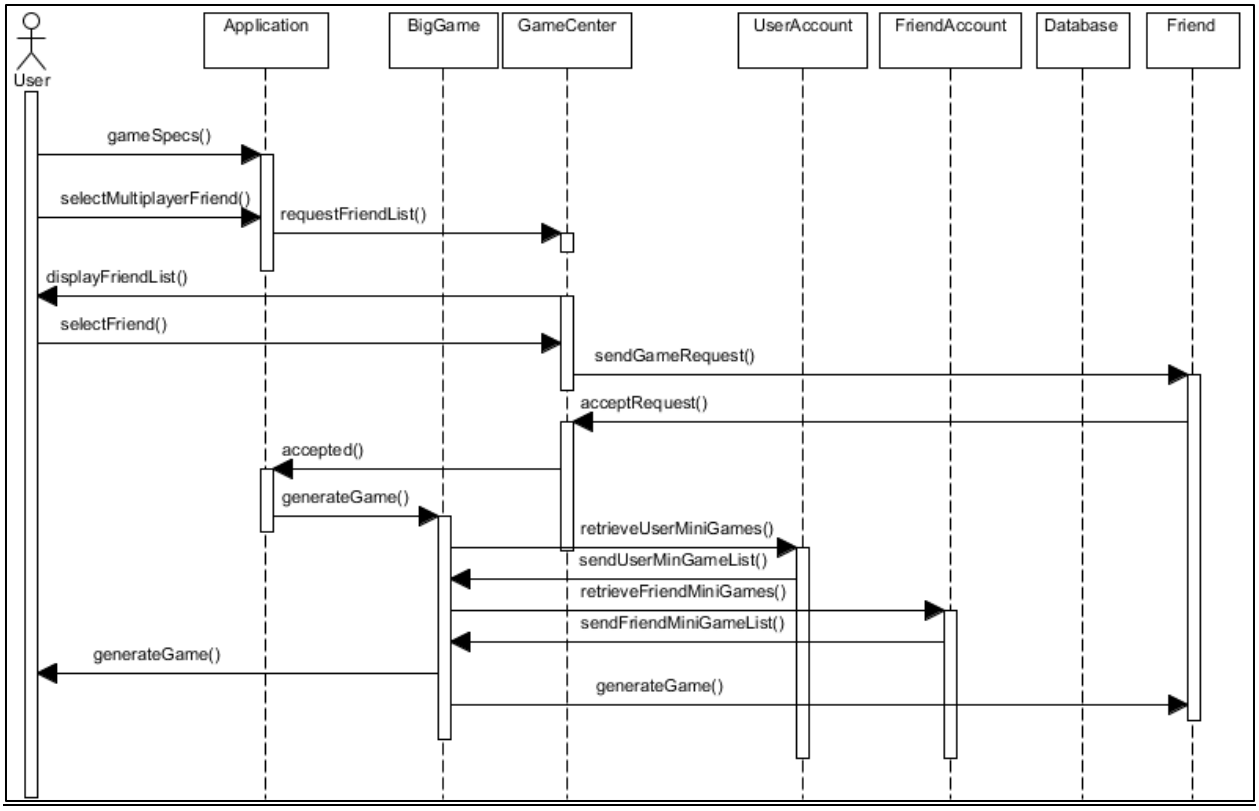


Figure 16*: Friend Gameplay Sequence Diagram*

State Diagrams

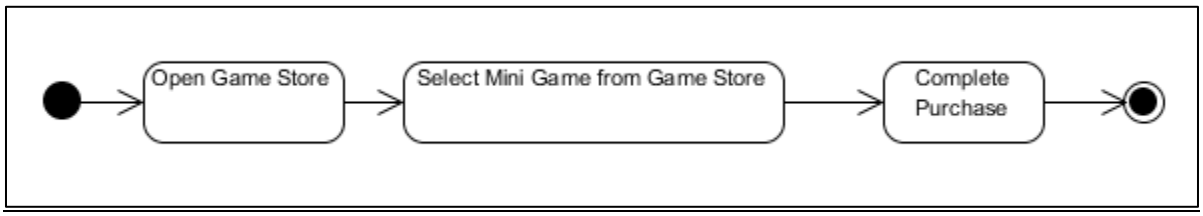


Figure 17*: Downloading Mini-Game State Diagram*

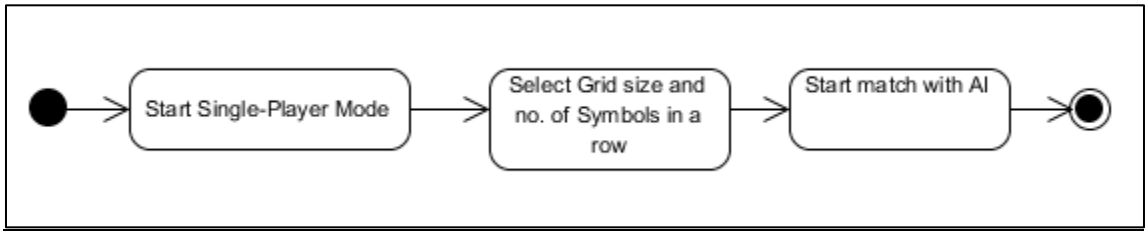


Figure 18*: Single Player Game Play State Diagram*

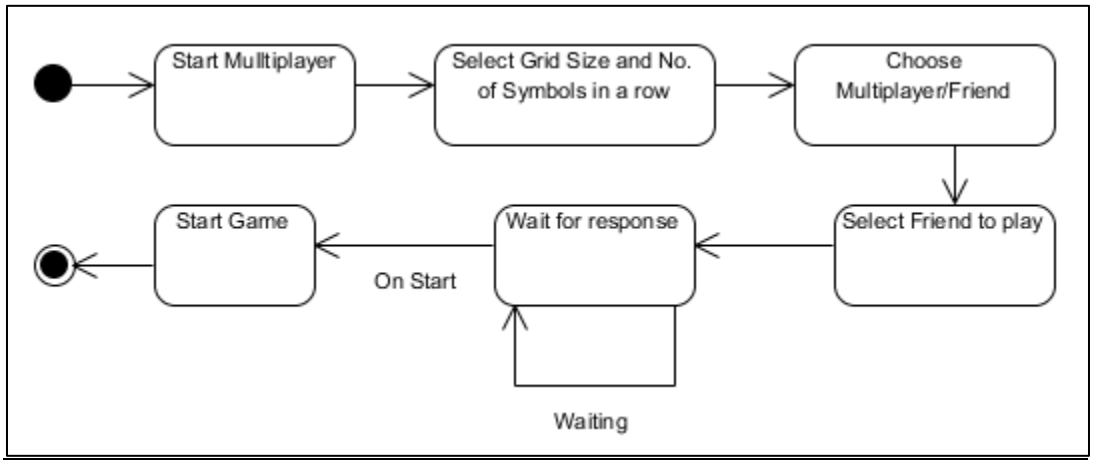


Figure 19*: Friend Game Play State Diagram*

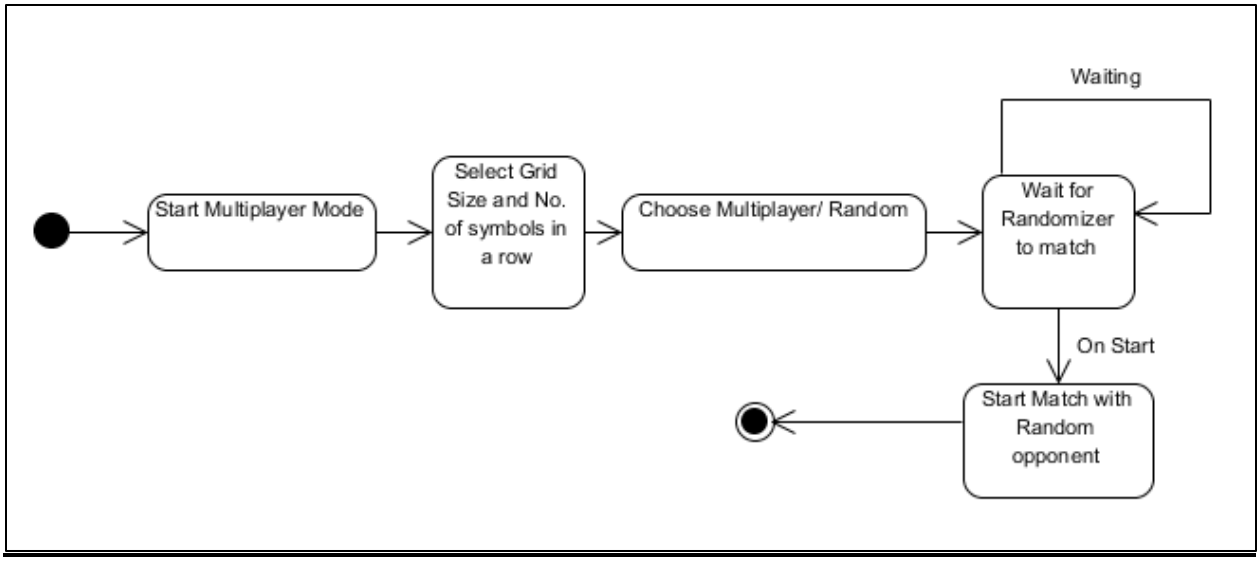


Figure 20*: Random Game Play State Diagram*

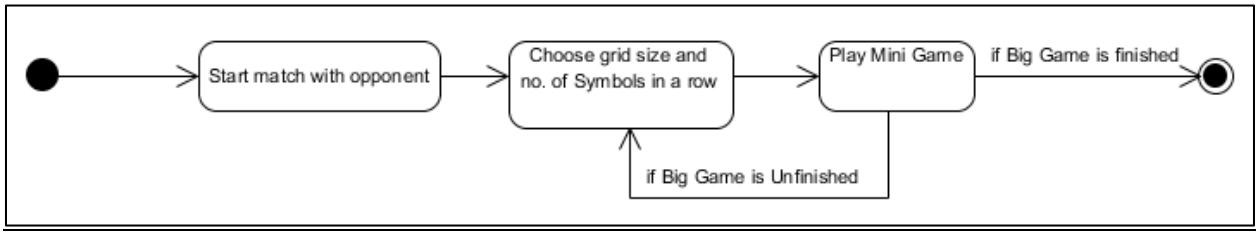


Figure 21*: Big Game Play State Diagram*

23 User Interface

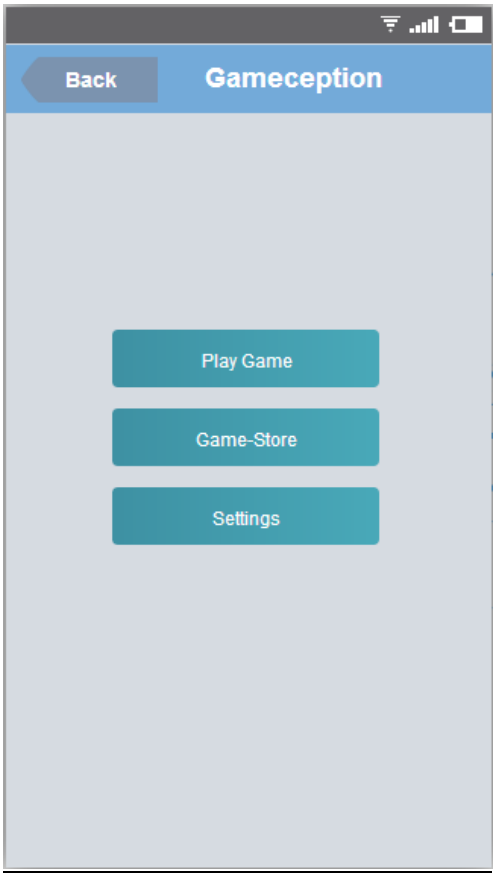


Figure 22:*The Main Screen of Game*

Here user has option to play the game, visit Game-Store or enter settings

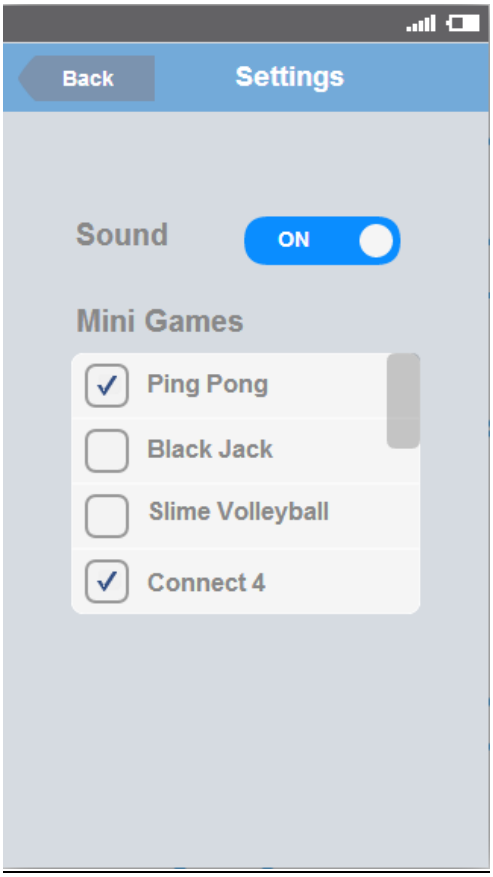


Figure 23*: The Settings Menu of the Game*

User can toggle the sound FXs and the mini-games which he would like to play within the big game.

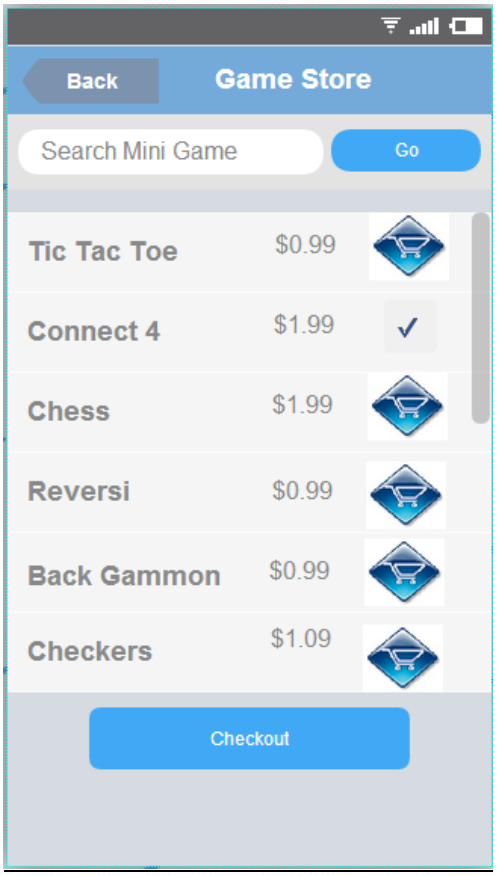


Figure 24*: The Game-Store*

User can buy Mini-Games provide by Third-Party developer.

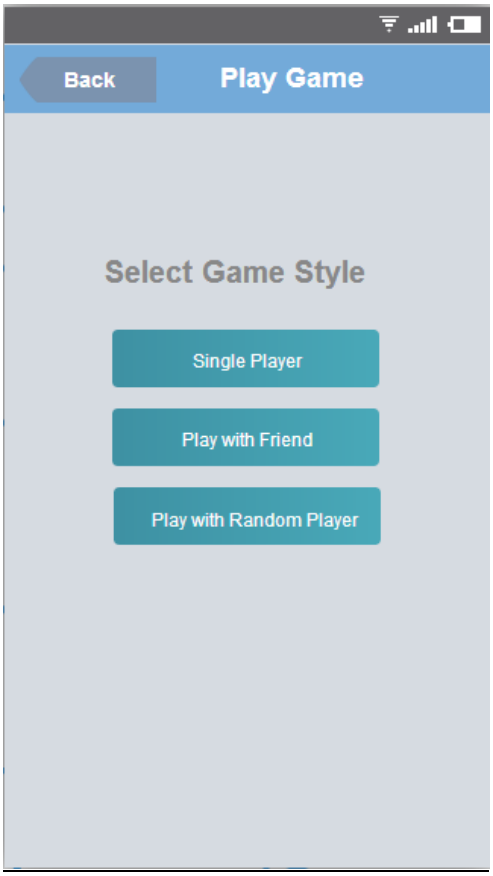


Figure 25*: The Main Screen of Game*

User when selects the play-game option, is showed player options.

Figure 26*: The pregame option*

User here has to select the grid size and no. of symbols in a row required to win the game.

Figure 27*: The Big-Game*

Grid consists of the Big-Game, where user may determine which location he want to play the Mini-game at.

III Design

24 System Design

24aDesign goals

Performance Criteria:

Design Criterion	Definition
Response time	Gameception should be able to handle user interface requests and responses immediately. The actions performed by the user should be reflected on screen without a delay.
Speed and Accuracy	Gameception would provide fast and accurate connections for multiplayer games. There should not be a lag between the user's actions being reflected on the opponents screen.
Capacity	The Gameception gameserver should be able to handle a large traffic when a large number of users are downloading minigames from the gamestore.

Dependability Criteria:

Design Criteria	Definition
Fault Tolerance	The game system should be able to handle user errors and network failure errors. In case of user errors the system notify player of error with a prompt. In case of network failure, system should backup all data and prevent data corruption.
Security	The game system should be secure from online attackers. Users' privileged details must remain secret and available only to the users themselves and the system administrator.
Reliability	The game should be reliable and the system should perform the appropriate response based on the user's actions
Availability	The game should be, ideally, available all the time, whenever the user wishes to use it.
Robustness	The game should be able to provide appropriate user feedback on encountering invalid user input

Cost Criteria:

Design Criterion	Definition
Development Cost	The development cost for the game would not go beyond the budget set for the game
Deployment Cost	The deployment cost would not exceed the preset amount
Upgrade cost	Upgrading the game to a higher platform or domain should include the cost of backward compatibility with the previous system
Maintenance cost	Maintenance cost for the system should not exceed the maintenance budget for the application

Maintenance Criteria:

Design Criterion	Definition
Updates	The users should be prompted to install updates that may be necessary for the application. The user should also have the option of auto-updating the document.
Game Maintenance	The game should be maintained by system administrators who are familiar with the game.
Access & Security	All player credentials would be stored in a secure database, with proper authentication protocols implemented for accessing the data
Portability	The game should be easily portable to a different platform than the current target platform (eg. From the current iOS and Android onto the Windows Phone)
Readability	The implementation code should be easy enough to understand and relate to the game for the administrator or maintenance engineers

End User Criteria:

Design Criterion	**Definition**
Usability	The UI should be intuitive and relatable to the user. It should be easy to learn even for a novice user.
Utility	The game should allow the user to perform the intended actions and give appropriate responses in its application behavior.

25 Software Architecture

25aSubsystem Decomposition

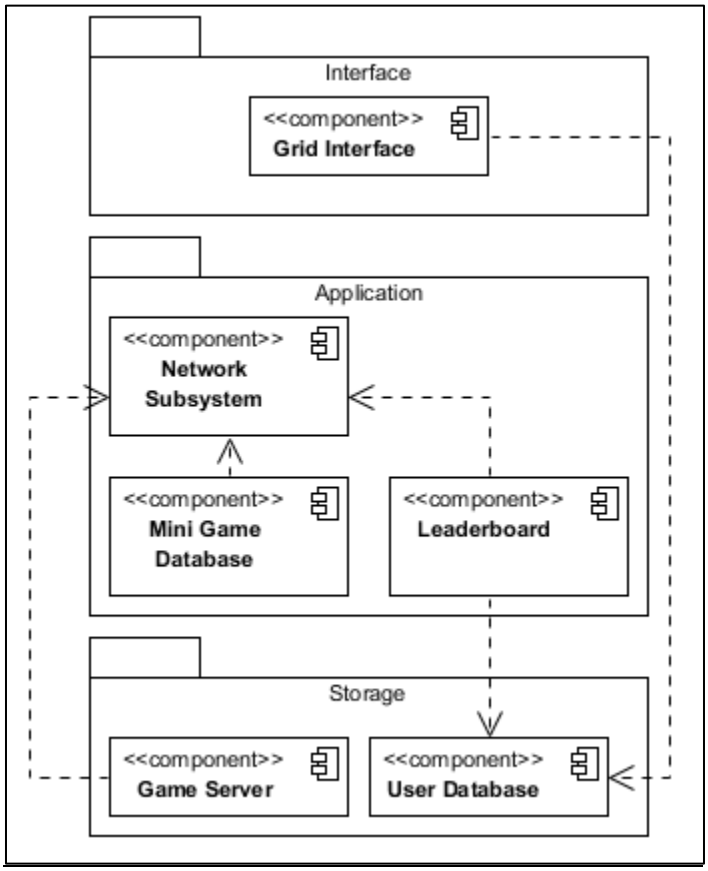


Figure 28*: Subsystem decomposition diagram(UML component diagram, layers shown as UML Packages)*

25b Hardware / software mapping

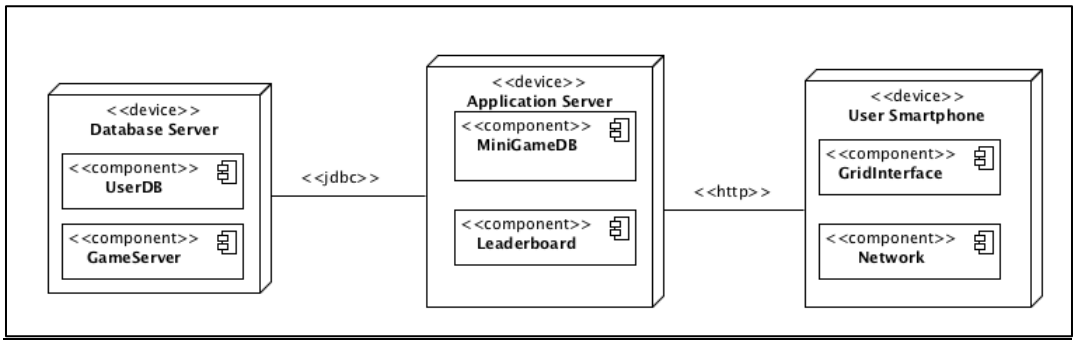


Figure 29*: Hardware and Software Mapping of Gameception*

25c Global software control

The Control flow is the sequencing of actions in the system.

The control flow paradigm this system is using is the event driven control flow. In event driven control, the main loop waits for an external event, which in our case would be the narrowing down of the random player or the successful location of a requested friend against whom the user wishes to play with; and when this event becomes available, the control is dispatched to the respective object, which in this case would be the random player or the requested friend who is then shown a request.

25d Boundary conditions

Use case name:	StartGameServer
Entry condition	1. The user logs into the Game Center using his account ID and starts Gameception
Flow of events	2. Upon successful login and startup, the user chooses the Game Store option to buy new Mini-Games. 3. The application connects to the Game-Server to retrieve the list of Mini-Games available to the user for download. If the connection is not completed, it prompts the user about the failure to connect
Exit condition	4. The Game Store displays the Mini-Games available for download to the user.

Use case name	StartLeaderboard
Entry condition	1. The user logs into the Game Center using his account ID and starts Gameception
Flow of events	2. Upon successful login the system requests the Game Center to retrieve the leaderboard for the application. 3. The user chooses to view the Leaderboard from the Game Center. If the Game Center fails to retrieve the Leaderboard, it tries to reconnect while giving a network error to the user.
Exit condition	4. The Game Center displays the Leaderboard to the user.

26 Subsystem services

UserDB - contains all the user information like name,display picture,account ID,wins,losses, owned mini games.

GameServer - This server stores all the mini games for the GameStore.

Leaderboard - As the name suggests,it contains all the ranking information w.r.t the whole game.

Network - It provides all the network layer services

GridInterface - This is the core user interface subsystem with which the user interacts

MiniGameDB - contains all attributes and information about the mini games.

27 Object Design

27a Object Design trade-offs

Delivery time vs. functionality

Delivery time is not a huge constraint on the system implementation. The functional requirements would gain precedence over delivery time during implementation.

Delivery time vs. quality

The quality of the product would gain precedence over the delivery time if there needs to be a compromise between the two. The delivery time is not as much of a constraint as maintaining quality requirements of the product.

27b Packages:

Databases

User Database: Holds all data relevant to the user. This includes userID, username, user profile picture, total wins and losses, a list of the user's friends, a list of the mini games the user has purchased, and a list of the mini games the user prefers to play based from the list of mini games he owns.

Mini-Game Database: Holds the data for all the Mini-Games that are available for download for the user. This includes the game name, cost, description, logo, popularity, number of downloads.

27c Class Interfaces

1. Class: Player
Description: Stores player information which is retrieved from the Game Center after proper authentication.

Methods	**Description**
authenticate	Attempts to log into Game Center with the player's Game Center e-mail address and corresponding password, if the user is not already signed in prior to starting the application. Ensures proper login to retrieve user information such as username, friend list, and statistics.

Constraints
`Context Player::authenticate() pre:` `  accounts->exists(a:Account | accounts.equals(a))`
`Context Player::authenticate(c:Credentials) post:` `  accounts.credentials = c`

2. Class: GameCenter
Description: Using the player's account information the Game Center retrieves the player's friend list. As well as using the player's total wins and losses to calculate a ranked leaderboard.

Methods	**Description**
sendFriendList	Retrieve the player's friend list using a Game Center call and store it in the user database as an array.
calculateLeaderboard	Retrieve the player's total wins and losses and calculate their rank based on assigning point values to the each win and subtracting each loss and then calculate their rank in comparison with the other Crimson players.
showLeaderboard	Display the calculated leaderboard of the top 100 Crimson players in numerical order starting with the player with the most points in first place. Also, display the player's rank even if he/she falls outside of the top 100.

Constraints
`Context GameCenter::sendFriendList() pre:` `  accounts->exists(a:Account | accounts.equals(a))`
`Context GameCenter::calculateLeaderboard() inv:` `  player.getWins < 0 | player.getLosses < 0`
`Context GameCenter::showLeaderboard() inv:` `  leaderboard.getSize() == 100`

3. Class: GameStore
Description: Handles the third-party minigames in app purchasing and installation.

Methods	**Description**
displayMinigames	Retrieve the list of third party mini games from the Minigame database and display the name of the minigame, cost, short description, logo, and statistics regarding how often the minigame has been downloaded and its popularity.
purchaseMinigame	Handles the in app purchasing of the minigame the user clicks on and confirms on purchasing using their Apple ID or Google Wallet information obstructed from the Gameception application.
installMinigame	Once the minigame purchase has been confirmed this method sends a request to the GameServer to begin the minigame download process to the player's device. Once this is completed the minigame must be installed and be added as a one of the available minigames to be chosen in the User's database and shown in the settings.

Constraints
`Context GameStore::purchaseMinigame() pre:` `  device.InAppPurchasingEnabled = true`
`Context GameStore::installMinigame() pre:` `  device.hasEnoughStorageSpace = true`

4. Class: Settings
Description: Gives the player the option to change their sound preference as well as their preference of which minigames they would like to play.

Methods	**Description**
setSound	Enables the system to either turn the default game sound on or off while the application is used.
setMinigamePrefs	Populates and displays the list of minigames the player owns using the user database and the minigamesPurchased attribute. Next to each minigame the system displays a checkbox to either enable or disable the minigame to be selected during gameplay as one of the minigames chosen in the grid.

Constraints
`Context Settings::setSound() pre:` `  device.hasSoundEnabled = true`
`Context Settings::setMinigamePrefs() post:` `  player.minigameNames.size() > 0`

5. Class: Gameplay
Description: Handles the basic gameplay aspects that are required to begin the main tic-tac-toe based game.

Methods	**Description**
getGridSize	The system retrieves the m x m size of the grid from the player's input prior to the start of the game.
getRowsForWin	The system retrieves the number of symbols required in a row for one of the player's to win the main tic-tac-toe game from the player's input prior to the start of the game.
getMinigames	The system retrieves the list of mini games to be randomly placed around the entire grid in each one of the m x m squares. This list is retrieved based off of each player's preference stored in their settings.
setGridSize	The system stores the grid size from the getGridSize method to be used in the GridInterface class.

Constraints
`Context Gameplay::getGridSize() pre:` `  Gameplay.gridSize() >= 9`
`Context Gameplay::getRowsForWin() pre:` `  Gameplay.RowsForWin() >= 3`
`Context Gameplay::getMinigames() pre:` `  player.minigameNames.size() > 0`
`Context Gameplay::setGridSize() post:` `  Gameplay.gridSize() >= 9`

6. Class: GridInterface:
Description: Displays the graphical interface for the main game.

Methods	**Description**
drawGrid	Draws and displays the entire grid layout (m x m squares) using the grid size attribute the user chooses before the start of the game.
displayMinigameLogo	Displays a mini game logo picture in each square on the grid corresponding to the mini game which was randomly selected for that game.
displayMinigameResult	After each mini game has been played the corresponding square will be updated to display an 'X' or 'O' symbol pertaining to the player who won that particular mini game. This symbol is placed on top of the mini game logo.

Constraints
`context GridInterface::drawGrid() pre:` `  player.gameplay.setGridSize()!= null`
`context GridInterface::displayMinigameLogo() pre:` `  minigame.getLogo() != null`
`context GridInterface::displayMinigameResult() pre:` `  minigame.returnResult() != null`

7. Class: Minigame
Description: Handles the result of each minigame to be sent to the main tic-tac-toe game.

Methods	**Description**
returnResult	The system returns a string corresponding to the username of the player or AI that has won the minigame to be stored and used by the GridInterface class.

Constraints
`Context Minigame::returnResult() post:` `  minigame.returnResult() != null`

8. Class: MinigameAI
Description: The AI will interact with the Minigame and play against the player.

Methods	**Description**
playMinigame	The Artificial Intelligence agent will play the minigame based on the set rules and definitions implemented by the minigame creators and taking a parameter of minigame name as a string.

Constraints
`Context MinigameAI::playMinigame() pre:` `  minigameAI.minigameName() != null`

9. Class: BigGameAI
Description: The AI will play the main tic-tac-toe game and strategically pick grid squares that favor a victory.

Methods	**Description**
pickGridSquare	The Artificial Intelligence agent will select a grid square to play the next minigame if they lost the previous minigame or if the game just started and they were randomly selected to go first. The AI will base their selection on finding a square in which they could easily use to win the game by placing a symbol in a row or selecting a square on the grid to prevent their opponent from winning the game.

Constraints
`Context BigGameAI::pickGridSquare() inv:` `  bigGameAI.turn().equals("AI") == true`

10. Class: MultiplayerRandom
Description: Matches two random players together in a game based on the same preferences and minigames.

Methods	**Description**
getRandomPlayer	The system will automatically try to match two players who have selected the same grid size, number of symbols required to win, and have the same minigames selected. If this match is not found the system will match two players based on having the same minigames selected. If this match is still not found the system will just match any two players together and start the game.

11. Class: MultiplayerFriend
Description: Matches two friends together in a game after one friend invites another to a play invoking the Game Center.

Methods	**Description**
displayFriends	The system will invoke Game Center to display a list of the player's friends from the User database.
getFriend	Retrieve the userID for the friend the user wants to invite to a game.
sendFriendNotification	Invoke Game Center to send the friend a game invitation and notify them of the impending game.

Constraints
`Content MultiplayerFriend::displayFriends() pre:` `  player.friendsList().getSize() >= 0`

`Content MultiplayerFriend::getFriend() pre:`
 	
`  accounts->exists(a:Account | accounts.equals(a))`

IV Test Plans

28 Features to be tested

1. Application launch,open and game-center Sign-in.
2. Game-store purchase.
3. MiniGame selection.
4. Gameplay with AI agent.
5. Gameplay with random player.
6. Gameplay with friend.

29 Testing materials (hardware / software requirements)

A. Hardware:
   i. Any iPhone later than iPhone 4.
   ii. Any iPad later than iPad 2.
   iii. Any iPad mini.
   iv. iPod touch 5th generation or later.

B. Software:
   i. iOS 4.1 or later.

30 Test cases

Test Case 1:

Test Case Name:	Open App and Game Center sign-in
Corresponds to:	Functional Requirement 9.1, Non-Functional Requirement 10.a.1
Pre-Condition:	Test application is launched by user. At least 3 testers are required successfully be displayed with the main screen, to pass the test
Flow of Events:	• Testers start the application. • "TU1, TU2 & TU3" should see the main screen of the application. • In the main screen there should be three available options each titled as: Play Game, Game Store and Settings. • A pop-up from the Game Center saying "Successfully Signed-in" is displayed.

Test Case 2:

Test Case Name:	User Game-Store
Corresponds to:	Functional Requirement 9.7, Non-Functional Requirements 11.d.2, 13.a.5 & 13.d.1
Pre-Condition:	Test User starts the application and wants to buy new games. At least 5 testers are required to buy Mini-Games from the Game-Store for this test.
Flow of Events:	• Tester click on the "Game-Store" button on main screen. • A new screen showing the list of available games at Game-Store is displayed. • Test user checks all the games he wishes to purchase. • Test user clicks on "Checkout" • User is confirmed for the purchase. • Game is now downloaded to mobile and visible on available Mini-Game list.

Test Case 3:

Test Case Name:	Select Mini Game
Corresponds to:	Functional Requirement 9.6, Non-Functional Requirement 13.a.4 & 14.b.3
Pre-Condition:	Test users clicks settings button and wants to make a change in his selected game list. At least 2 users should be able to successfully update his selected game list
Flow of Events:	• Test users TU1 and TU2 click on the settings game option. • Test users are displayed the list of Mini-games they own with a selectable checkbox to the right. • Test users select the Mini-Game they wants to play in the Big-Game • They then click back button. • The selected options of the list are updated. • Test User click on the settings button again and find the changes in the list reflected successfully.

Test Case 4:

Test Case Name:	Play game option page is displayed
Corresponds to:	Functional Requirement 9.3, Non-Functional Requirement 16.b
Pre-Condition:	Test user has selected ‘Play game’ option from the main screen. At least 5 user need to successfully be displayed with game option page for the test to succeed.
Flow of Events:	• Test user TU1, TU2, TU3, TU4 & TU5 select play game option. • Test users are shown a new page. • In the screen there should be three available options each titled as: Single Player, Play with Friend and Play with Random Player.

Test Case 5:

Test Case Name:	Play with AI agent
Corresponds to:	Functional Requirement 9.3, Non-Functional Requirement 10b
Pre-Condition:	Test users have selected Play Game option. At least 5 Test Users are required for this test.
Flow of Events:	• Test users selects Single player from the play-game options screen. • Test users is asked the size of the grid and no. of symbols in a row to win in a dialog box. • Test users inputs appropriate values and clicks 'next'. • Based upon selection of Mini-Games by the player, the Gameception grid is loaded with Mini-Games to play. • Test users now plays the Big-Game with an Artificially Agent.

Test Case 6:

Test Case Name:	Play with Friend
Corresponds to:	Functional Requirement 9.3 and 9.8, Non-Functional Requirement 13.b.4 & 10.c
Pre-Condition:	Test user has selected Play Game option. At least 2 Test Users are required for this test. Test user TU1 and TU2 are friends with each other.
Flow of Events:	• Test user TU1 selects Play with Friend from the play-game options screen. • Test user TU1 is asked the size of the grid and no. of symbols in a row to win in a dialog box. • Test users inputs appropriate values and clicks 'next'. • Test User TU1 selects the friend to play the game with. • Test User TU2 receives a play-game request notification from TU1. • Test User TU2 accepts the request. • Based upon selection of Mini-Games by both the players, the Gameception grid is loaded with Mini-Games to play. • Test users TU1 and TU2 now play the Big-Game with each other.

Test Case 7:

Test Case Name:	Play with Random Player
Corresponds to:	Functional Requirement 9.3 and 9.8, Non-Functional Requirement 10.c & 13.b.4
Pre-Condition:	Test user has selected Play Game option. At least 5 Test Users are required for this test. Test user TU1, TU2, TU3 , TU4 and TU5 are not friends with each other.
Flow of Events:	• Test users viz. TU1, TU2, TU3 , TU4 and TU5 selects Play with Random Player from the play-game options screen. • Test users TU1, TU2, TU3 , TU4 and TU5 are asked the size of the grid and no. of symbols in a row to win in a dialog box. • Test users TU1, TU2, TU3 , TU4 and TU5 inputs appropriate values and clicks 'next'. • Test Users TU1 & TU4 are then matched with each other on the basis of size of the grid and no. of symbols in a row to win and Mini-Game preference. • Test Users TU1 and TU4 receives a match found notification from Game Center. • Based upon selection of Mini-Games by both the players, the Gameception grid is loaded with Mini-Games to play. • Test users TU1 and TU4 now play the Big-Game with each other.

Test Case 8:

Test Case Name:	Toggle Application Sound
Corresponds to:	Functional Requirement 9.10, Non-Functional Requirement 16.a
Pre-Condition:	Test users clicks settings button and wants to mute the sound.
Flow of Events:	• Test user clicks on the settings game option. • Test user is displayed with a toggle switch for sound (on -off). • Test users clicks on the toggle switch, which turns to Off state. • They then click back button.. • Test User can no more hear the sound from application..

31 Testing schedule

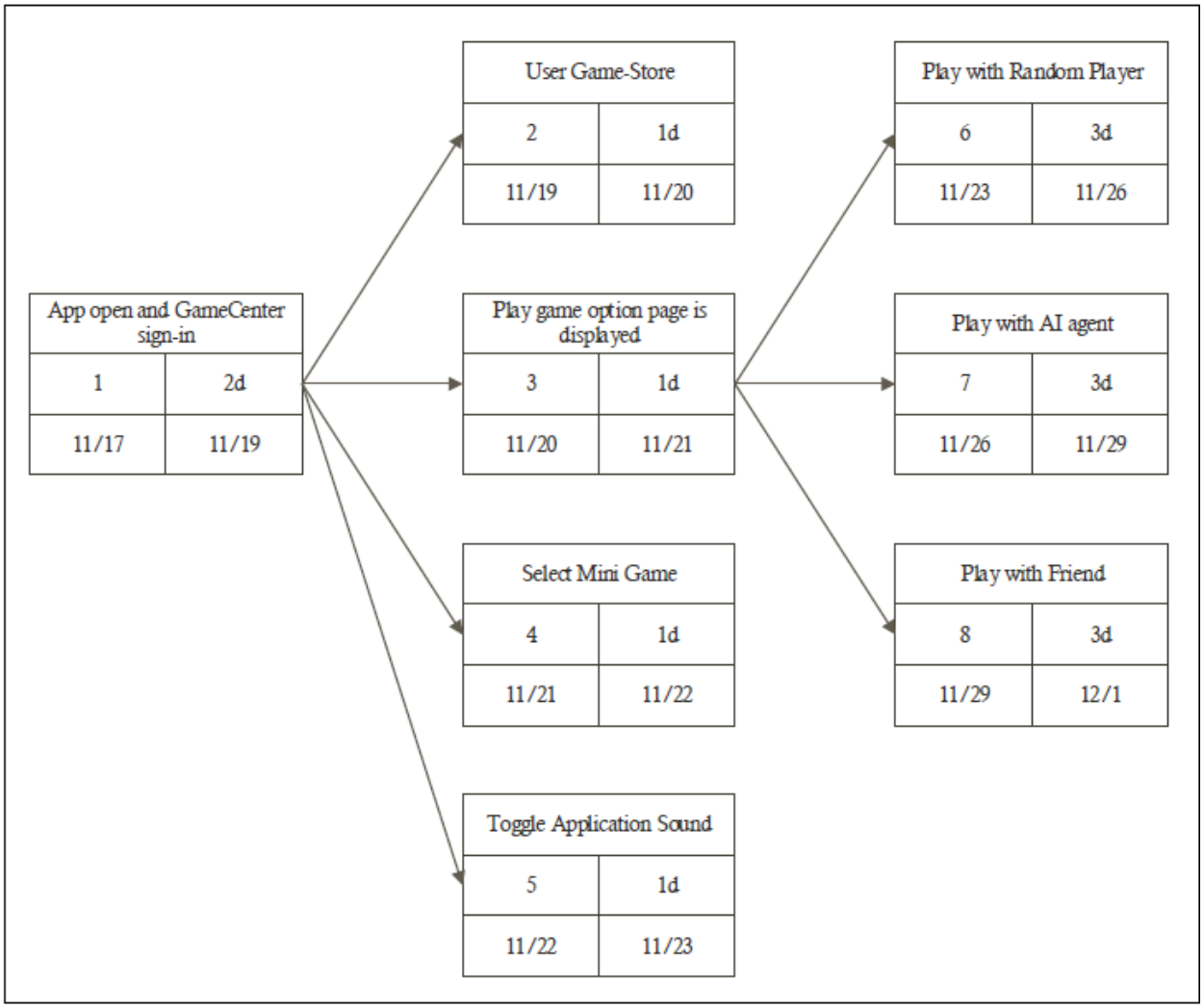


Figure 30: *PERT Chart*

V Project Issues

32 Off-the-Shelf Solutions

32a Ready-Made Products

GameCenter API - The GameCenter API will be used to display leaderboards and will also be used for the random match.

32b Reusable Components

The Apple Gamekit Framework Reference can be used to build mini-games as a reusable component.

32c Products That Can Be Copied

Existing games can be ported to comply with our mini-game requirements.

33 New Problems

33a Effects on the Current Environment

Ideally, since this is a game application developed using the Android or iOS standards, there should not be any adverse effects on the User's device. However, the implementing engineers must make sure that the application does not overburden the device's processor.
Precaution: The application performance and memory management must be optimized to avoid any adverse effects on the user device's processor.

33b Effects on the Installed Systems

The application should only access the required privileges on the user's device so that it can function. It should not be able to modify any content within the device that is not related to the application.

Precaution: The user must be prompted before the application gains access to privileged permissions within the device. For example, the application may use the user's contact list to retrieve the user's friend contacts and allow better matching for the Game Center. The project supervisors must ensure that no malicious code is included within the application code itself.

33c Potential User Problems

The user could face problems of memory management and slow device response time. The user could also face a slower response time from the application itself, if it has not been implemented properly.

Precaution: The application performance and memory management must be optimized to avoid any adverse effects on the user device's processor.

33d Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

The projected application is planned to be easily conceivable within the Anticipated Implementation environment.

The limitation that may exist, if at all, would be the fact that Android does not have an equivalent of the Game Center to allow games with friends and auto-matching. However, 'Play Games' application from Google should be sufficient enough to implement these features for the android devices.

33e Follow-Up Problems

There may be a few problems that may be faced by the product which cannot be anticipated in the current context:

- If the application stores for Apple and Google change their policies in a significant manner, the product may have to be altered to match those criteria.
- If there is a scenario wherein there is a sudden decrease in popularity of hand-held devices (although unlikely, going by the current trends), the game would have to be ported to the most popular platforms available at that time
- If there are any new changes made to laws in particular countries or globally, that may affect the application or design, then the product needs to be modified so that it is not in violation of any such laws.

34 Tasks

34a Project Planning

The development cycle will consist of two release phases each consisting of two sprints that will each have duration of approximately two weeks each, totaling a two month span to complete the entire application.

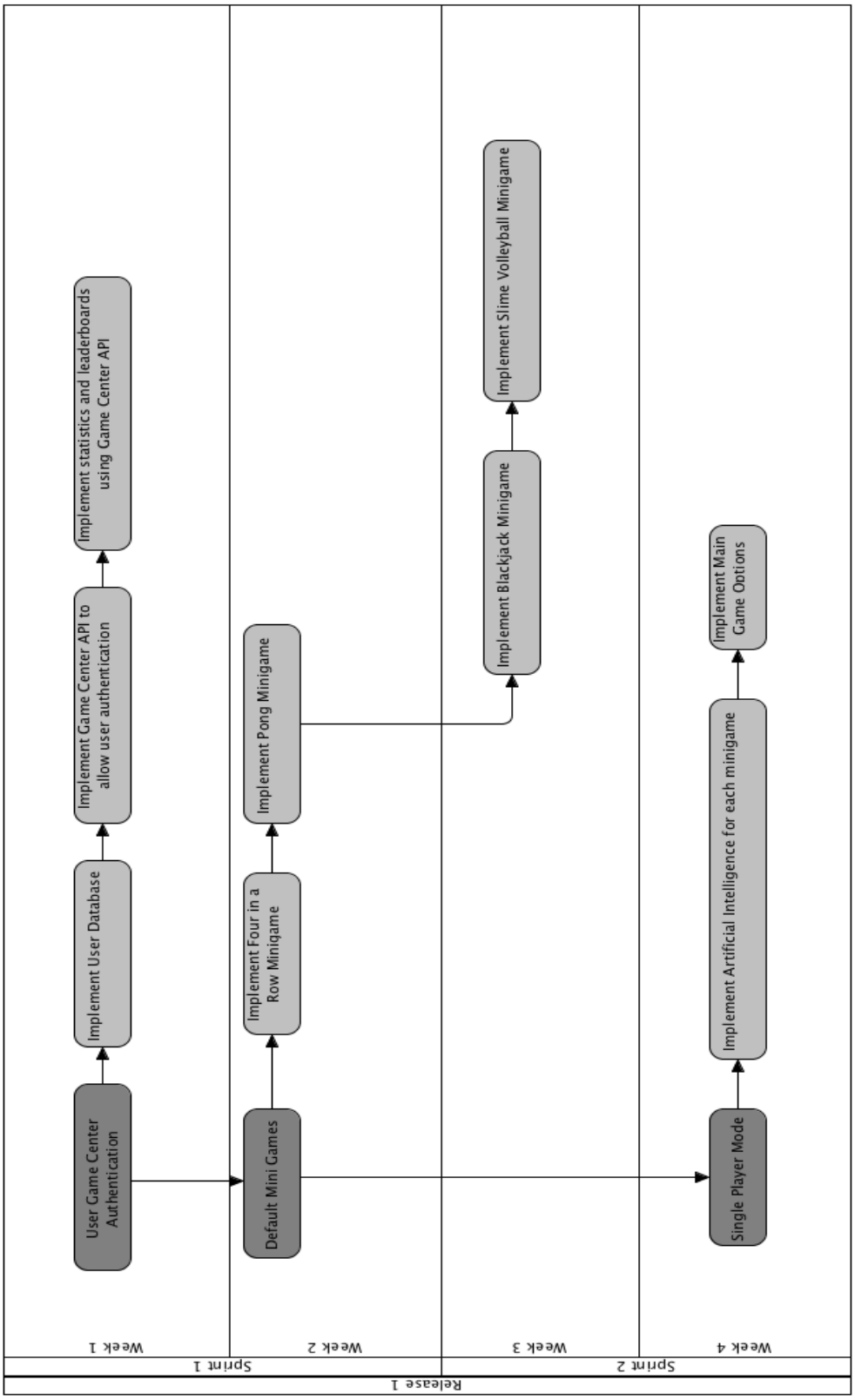


Figure 31: *Release 1 Diagram*

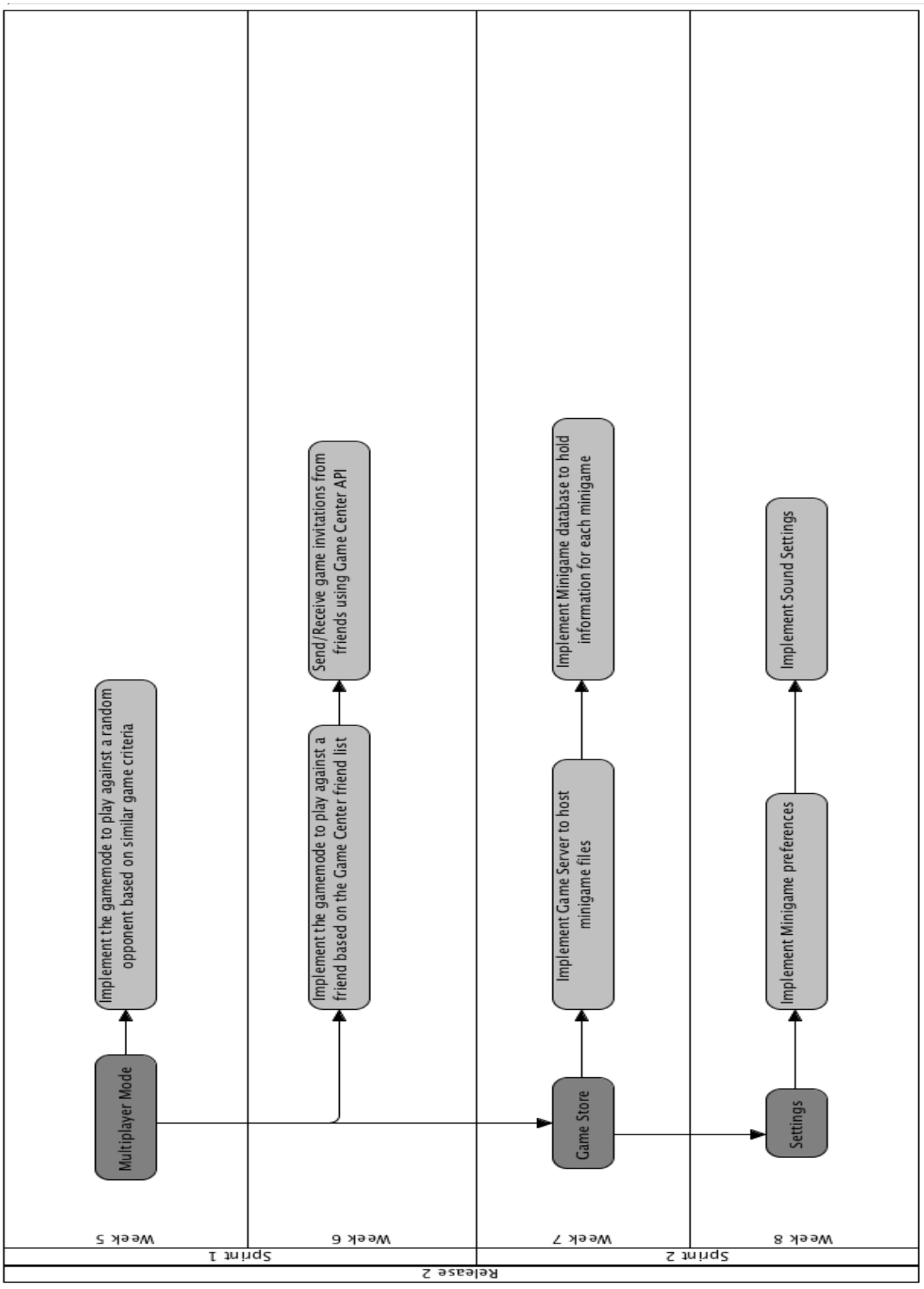


Figure 32: *Release 2 Diagram*

34b Planning of the Development Phases

Phase	Required Operational Date	Operating Environment Components	Functional Requirements
Implement User Game Center Authentication	Release 1 Sprint 1 Week 1	Implement User Database.	The player should be able to use the application using his Apple ID and Game Center account. The player should be able to view his own statistics as well as view a leaderboard with the top players.
Implement Default Mini Games	Release 1 Sprint 1 - 2 Weeks 2 -3		The application should include the following four mini-games by default: 4-in-a-row, pong, blackjack, and slime volleyball.
Implement Single Player Mode	Release 1 Sprint 2 Week 4	Implement Artificial Intelligence	The application should allow the user to play against an artificial intelligence agent. The application should allow the user to select the size of the grid, with the minimum grid size being 3 x 3. The application should allow the user to select the number of symbols in a row required to win a game, the minimum being 3 and the maximum being the number of grid rows the user selected.
Implement Multiplayer Mode	Release 2 Sprint 1 Weeks 5 - 6	Implement multiplayer matchmaking.	The application should allow the user to play against a friend using the GameCenter friends list or against a random opponent. The player should receive notifications when another player wants to play with them through the Game-Center.
Implement Game Store	Release 2 Sprint 2 Week 7	Implement Minigame database and Game Server	The player should be able to download third-party games from the native Game- Store.
Implement Settings	Release 2 Sprint 2 Week 8	Add column to User Database table corresponding to the Mini Games the user prefers to play.	The player should be able to choose which mini games can be played, and whether they would like to turn in game sounds on or off.

35 Migration to the New Product

35a Requirements for Migration to the New Product

The application would not be building upon an already existing game. Therefore, there should not be any requirements based on a previous downloadable module. Since the application would be available in the Apple and Android application store. There would be a few minimum requirements for the product to be run on the user's device:

- The user's device OS must satisfy the intended OS level for the application.
- The user must have the memory space required to install the application.
- If there is an update to be pushed to the application the user must have enough memory space on his/her device to install the update.

35b Data That Has to Be Modified or Translated for the New System

The game would not require any new translations or modifications to be implemented onto the user's device. The game would be installed as a separate module from the application store and will not modify any of the already existing applications in the device. The only exception to this case would be during updates to the product, wherein the module which is being updated would be altered by the update file downloaded.

36 Risks

Based on the requirements of the project, the developers may face a few risks. These may include but are not limited to the following:

1. Server Overload: Based on the popularity of the game, it is possible that the developers may underestimate the number of users of the application. Therefore, the servers hosting the database and the central application may be overloaded.
   Solution: The developers must make sure that there is enough buffer to accommodate a surge in user traffic

2. User Memory Limitations: The game requires the user to download mini games from the Game Store. Every download would take up space on their device, depending on how big the mini game is. Therefore, the users run the risk of exhausting their memory due to just one game.
   Solution: The administrators and mini game approvers must make sure that the sizes of the mini games are small enough so as not to exhaust the user's memory. The mini game developers must also take that under consideration.

3. Low Quality: Since the mini games would be included by third-party developers, there is a risk of poorly created or low quality games being introduced to the Game Store.
   Solution: There needs to be a quality criteria set for the mini games that would make it to the Game Store

4. Low Productivity: One of the USPs of the game is that there are different games that can be played within it. The developers can face the risk of the non-addition of newer mini games.
   Solution: The third-party developers must be provided with proper incentives to encourage creation of newer mini games.

5. Schedule Pressure: The developer team might face issues in meeting the deadlines set by the organization for delivering the project.
   Solution: The team must organize and plan for any setbacks they might face and take any potential errors under consideration while planning the project implementation

6. Silver Bullet Syndrome: The developers can face the risk of becoming over dependent on a particular patch of fix for the application and put all their efforts in to it while believing that it would solve their problems.
   Solution: The developers should avoid this attitude and try to explore different methods of achieving a breakthrough.

37 Costs

Monetary Costs:

i. Development Kit: The Development kit units need to be bought for all the hired developers. The kit would include a MacBook Pro computer (approx. 2,000$),a developer license (99$) and the documentation (free.)
ii. Web Server: We need to configure and build a web server to host our minigames and other data. This would cost approximately 1,000$.
iii. Employees: Employees need to be paid. This cost would depend on a lot of factors like employee skill, location etc.

Time-based costs:

i. Web Server configuration: This would take approximately a week.
ii. Web Server development: This would take 4 months depending on the skill and availability of the developers.
iii. Web Server maintenance: This would be a recurring indefinite task.
iv. App development: This would also take 4-6 months depending on the number and skill of the developers.
v. Third party developer reference: This would take about a month.

38 Waiting Room

1. **Number of Users**: The number of users that can be handled at a time could be increased to 20,000.
2. **Parallel downloads**: The user should be able to download multiple mini games concurrently.
3. **Better Graphics:** With the increasing availability of GPUs (Graphical Processing Units) in modern smartphones, higher graphical affordances can be given to the proposed mini games.
4. **n-Player gameplay**: More than 2 players could compete in a single instance of a Big Game.

39 Project Retrospective

In retrospect, the development process has created a viable game that can be considered as a profitable entry into the mobile gaming industry, which was a primary requirement for the organization while commissioning the team to create the game. The game emphasizes on multiplayer gaming with a social aspect, which the team believes is an apt application for the current market.

The Initial Phase
Coming up with an idea for a game application based on a traditional table-top game was a challenge in itself. The game had to be original and engaging enough to be played on a regular basis.
The idea of Gameception stemmed from the intention to integrate this traditional aspect with a touch of modernity that would be befitting a modern mobile device such as a smartphone or a tablet.

Requirements
The requirements that were provided by the design team were carefully constructed by considering the various subsystems that would be involved in the creation of such a game and the actors that would be involved in using these subsystems. Every conceivable situation that could be thought of during the implementation and the actual usage of the game played a role in deciding these requirements.

System Design
With respect to the system design, an outtake from the Gameception application is the Auto-match option in the multiplayer game. Using the auto-match the GameCenter matches a user with other users using the application that have more games in common. The Leaderboard was another feature that was key in the system design to keep a track of user performance compared to other users of the application.

What Worked Well
The collaboration and teamwork of all members was vital in creating the game. Even though there were differences in opinions during the project's infancy, the team managed to get a consensus on the final product.

Glossary

Device: The users iOS or Andriod phone or tablet

Game Center: An online multiplayer social gaming network that allows users to invite friends to play a game, start a multiplayer game through matchmaking, track their achievements, and compare their high scores on a leader board. It is implemented by the device itself (e.g. iPhone, iPad, etc).

Game-Store: An embedded store within the game that will be used to download new games to play inside the tic tac toe grid. This is independent from the native app store associated with the user's device.

Grid: The main frame consisting of a 3 X 3 matrix of squares in which the game is played.

Square: Individual sections of the grid which contain embedded games.

Mini-game: the embedded games within the squares.

Symbol: A symbol is either an 'X' or an 'O' on the grid.

References / Bibliography

[1] Bernd Bruegge and Allen H. Dutoit, Object-Oriented Software Engineering using UML, Patterns And JAVA™.

[2] http://mathwithbaddrawings.com/2013/06/16/ultimate-tic-tac-toe/