Detailed Design Report

for

Building a Client-Server Architecture to Play Card Game, **BLÖFLÜ PİŞTİ** , from Mobile Devices

e-Limon

A.Emirhan ÖZDEMİR

Hamza YILMAZ

Utku SAKİL

Cuma Tekin TOPUZ

Contents

1.Introduction

This document is a detailed design report for **Building Server-Client Architecture to Play Card Games, BLÖFLÜ PİŞTİ, from Mobile Devices.** To introduce you the document well, we will first give problem definition of our project and then the purpose of this IDR, scope of this document, then follow an overview which explain how the DDR is organized and what the report contains and we will state the descriptions and abbreviations that are used in our DDR and references which state all the documents and sources that are used in our DDR report. After finishing the introduction we supply **a System Overview** which constitutes of a general description of the software system including its functionality and matters related to the overall system and its design. We will briefly explain the goals, objectives and benefits of our Project in this part. Namely this part will provide the basis for the brief description of our product After completing **System Overview**, we will address **Design Considerations** which Special design issues need to be addressed or resolved before attempting to devise a complete design solution. Following **Data Design** which consists two separated part: **Data Description** that explain how the information domain of our system is transformed into data structures and the other **Data Dictionary** which alphabetically list the system entities or major data along with their types and descriptions . Then **System Architecture** and then **User Interface Design, Detailed Design, Libraries and Tools, Time Planning** and finally **Conclusion.**

1.1 Problem Definition

The aim of this project is to create a generic server-client architucture that enables users to play card games from mobile devices. As a sample case we chose to make that project with BLÖFLÜ PİŞTİ and Android OS. Creating a generic server-client architecture will provide us to reuse this system in a different game. Just making necessary modifications makes the system available for different card game.

BLÖFLÜ PİŞTİ played as a single deck of cards and without Jokers. Each player is dealt 4 cards. At the begining of the game, 3 of cards closed and 1 is open card are distributed on the floor. If players do not have any cards, 4 cards are dealt again to each player. The cards on the floor can be won by throwing the same card number with the top of the stack or throwing Joker. When the stack has only one card, then a player throw the same number he will get Pişti and get 10 points. Bluff option contributes to game an excitement. When a player make bluff, he throw the card closed. If opponent says this is bluff and he is right, opponent gets 10 points. Otherwise, player gets 20 points.

1.2 Purpose

The purpose of this document is to show how the software system will be structured to satisfy the requirements **Building Server-Client Architecture to Play Card Games, BLÖFLÜ PİŞTİ, from Mobile Devices** to be developed, it will contain all the information required by a programmer to write code. Any programmer can write all the core codes while reading this report because this developer can see all the requirement information about our project and the diagrams will show the way how developer starts coding and what he will code.

1.3 Scope

Scope of DDR is to provide information about detailed design of the project. This document covers the architectural design, data design, procedural design, design constraints, development schedule. Also hardware and software requirements and working environment will be explained.

1.4 Overview

This DDR consists of ten parts. In first part we gave general information about **Building Server-Client Architecture to Play Card Games, BLÖFLÜ PİŞTİ, from Mobile Devices** to the readers, because it is written for developers we gave all the required information in this report and developers can read easily this report and start coding. In second part of this report we focus on **System Overview** which provide a general description of the software system including its functionality and matters related to the overall system and its design. In the third part **Design Considerations,** we mention about special issues and then we give all information about **Data Design** which explain *Data Desctiption*, clarify how the information domain of our system is transformed into data structures and *Data Dictionary*, Alphabetically list the system entities or major data along with their types and descriptions. In the fifth part **System Architecture** which mention about architecture of the project. In the sixth section **User Interface Design** is the header and we will decsribe the functionality of the system from the user's perspective . It display screenshots showing the interface from the user's perspective and there is discussion of screen objects and actions associated with those objects. In the seventh part of our report the heading is **Detailed Design** which contains the internal details of each design entity/component. The last three parts are **Libraries and Tools, Time Planning and Conclusion.**

1.5 Definitions and Abbreviations

DDR: Detailed Design Report
AI bot/agent : Artificial intelligent game client bot/agent.
ACA: Android Client Application
GUI: Graphical User Interface

1.6 References

[1] IEEE Std 830-1998: IEEE Recommended Practice for Software Requirements Specifications
[2] http://market.android.com
[3] http://itunes.apple.com/us/app/pisti-ii/
[4] http://creately.com
[5] http://wikipedia.org

2. System Overview

Our system, **Building Server-Client Architecture to Play Card Games, BLÖFLÜ PİŞTİ, from Mobile Devices,** consists of three major parts namely server, client application, and the database. Server is responsible for managing all the interactions between clients, executing the game rules, running the AI bot and recording necessary information to the database. The client application is an Android application which has an user interface showing the game environment. Client application is responsible for taking input from users and send them to the server. Finally, the database is responsible for keeping information about user information, rankings, and statistics about the users.

2.1 Goals

The main goal of our project is to create a server-client architecture to develop card games. The minor goals of this project are developing a cute graphical user interface for the Android client application, an intelligent AI bot that can play against to every user differently.

2.2 Objectives

Our objectives about the server is to make it stable and fast enough. Meaning in fast enough is that this is human playing game environment so it does not have to be really fast. A user can play at most 1 move in 2 seconds. It is nothing compared to current computer and network technologies. Another objective is the one with the client application. It need to require reasonable memory and computing power because the game platform is mobile and it is fact that mobile platforms do not have so much memory and cpu source. And the final objective is to make the AI bot as intelligent as it can be. Of course intelligent is not an objective word but it has to play at least like regular human player. The advantage of the AI bot is that it never forget moves but human can. Disadvantage of AI is bluffing part, AI cannot know and anticipate when it should bluff or the opponent is bluffing. That is the part our project will be a solution to this problem. Using statistics in the database bluff handling will be done.

2.3 Benefits

This project is an entertainment project which is a type of card games, so the main purpose is to enjoy target people. Satisfying the requirements of Bloflu Pisti System will be enough condition to ensuring the entertainment level of game, because the Bloflu Pisti is a game that is ported from real life to digital media (software). Lastly, a slow reacting game cannot be enjoyable, then game software should be enough of quickness for running in an Android device. Our software will work very fast because user communicate with server instead of with each other.

3. Design Considerations

Building Server-Client Architecture to Play Card Games, BLÖFLÜ PİŞTİ, from Mobile Devices project is a card game on ANDROID mobile devices. This system works on a server and answer client request .

3.1. Design Assumptions, Dependencies and Constraints

3.1.1. Design Assumptions

The target group for this software is assumed as everyone who needs to be entertained. Players of this game software can play either with AI or with other users matched from multiplayer server architecture. Player number is allowed for zero to two players, that is, multiplayer servers with zero human player also can play AI versus AI in an ongoing game. Players can communicate via our servers' network protocol. Player requests the information from the server and server responds to players' request. User can play with an AI Bot or play with others. Players communicate with our server and our system answer their request and system take place of any players who log out the system immediately.

3.1.2. Design Dependencies

The client application software is considered to run on the Android operating system connected to a web server. Source codes of project will be implemented on ECLIPSE with Android Java Plug-In. In addition software will be running also on a web server. The server will be implemented on PHP.

3.1.3. Design Constraints

-Hardware Constraints

System will work an all new generation Android Mobile devices such as mobile phones with Hsdpa, touch tablets, hand PDAs and Android netbooks. Backward compatibility will probably be available up to Android 1.6.

-Software Constraints

We will implement the project on ANDROID SDK Eclipse Platform and the database of project will be created with MySQL, in addition Server will be created on PHP. The communication protocol will HTTP and the user login system will be Basic HTTP Authentication. Therefore JAVA, SQL, PHP are mainstream infrastructures for developing our project.

3.2. Design Goals and Guidelines

Primary aim of the software is to make a well infrastructed software with modular software architecture. Server-Client architecture will be well organized with other components of project. We intend that server-client architecture run in a generic way, so that possible changes in design patterns will not highly affect the outline of the project. In addition to this expectations, our software will include some other properties to be a successful software. Performance is important and it will be provided within the project, because android devices have relatively less computation power when comparing with notebooks or desktop computers. In addition simplicity should be embedded in a sophisticated design, so that user

can easily manage the content of a complex design. Lastly server will be tweaked to refresh the users' move interrupts with a rate of half second, that is, client application will try to reach server with a frequency of 2 Hz.

In general, we have several design guidelines that we take into account:

1. Useful: Result of the project will be useful for everybody.

2. Fast: Software product should run with no delay time. Even every millisecond is important nowadays. AI agent will be quick enough to respond against human players' moves and in case of an connection interrupt from player side, an AI agent will take place with reasonable delay.

3. Simple: Keeping design and user interface as simple as possible

4. Engaging: Once a person uses that product, s/he should want to try again, so it must be a challenging task to do.

5. Universal: Design should be something that covers principles which have standards around the world. Design standarts should also be satisfied, and generic mainstream properties should fulfill globally accepted expectations.

6. Attractive: User interface or graphical components of design should be attractive to people, delight the eye without distracting the mind.

-Software System Attributes

Usability: The system should have easy user interface with minimal design. and minimum number of user interfaces. Any person with the knowledge of basic computer usage should be capable of using our system.

Documentation: The system should include a tutorial-like documentation which includes the information about how to play game and use the application.

Availability: The system should be available to user any time s/he wants to access with his/her username and password

Reliability: The system works independent from the users namely user communicate with server and server responds the user. If a user wants to make a move s/he sends the request to the server and then system do the action then responds quickly. There is tolerance of that any user leave the system immediately.

Security: System works in a secure atmosphere and it does not let intruders. Passwords of users are kept secure and sent with an encryption algorithm which is secure enough to use.

4. Data Design

4.1. Data Description

This section of the document contains a description of the information domain of the data entities of this project.

4.1.1. Server Data Entities

This section describes the data entities of the server part of the project.

4.1.1.1.User Data Entity

This data entity refers to each user in the database.

Field Name	Data Type	Description
userID	integer	unique id of the user
email	string	email address of the user
username	string	username of the user
password	string	md5 hash of the password

4.1.1.2.Card Data Entity

This data entity represents each of 52 cards in the game.

Field Name	Data Type	Description
number	integer	number of the card [1,13]
kind	integer	which kind is the card [1,4]*
belongs	integer	reference to owner player id

** 1=>Clubs 2=>Diamonds 3=>Hearts 4=>Spades*

4.1.1.3.Player Data Entity

This entity represents the data of each player online at the system.

Field Name	Data Type	Description
player	User	user information of the player
sessionID	long	unique session id of the player
cards	Card[4] array	list of cards that player has

4.1.1.4.GameTable Data Entity

This data entity refers to each game table on the server at that time.

Field Name	Data Type	Description
tableID	integer	unique id of the table
tableName	string	name of the table
players	Player[2] array	2 opponent players
onFloorCards	Card[4] array	list of cards on the floor
state	GameState	state of the game at this table

4.1.1.5.GameState Data Entity

This data entity represents state of the game on each table.

Field Name	Data Type	Description
cards	Card[52]	list of all cards at this game
state	Macros	state of the game
turn	integer	which player has turn 1 or 2

4.1.1.6.GameMove Data Entity

GameMove data entity represents the move that every player makes.

Field Name	Data Type	Description
player	Player	which player made the move
isOpen	boolean	card is thrown as open/close
card	Card	which card is moved

4.1.1.7.Statistics Data Entity

Statistics data entity keeps the information from the database.

Field Name	Data Type	Description
player	Player	statistics of which player
bluffRate	double	player's bluff rate in past games
successfullBluffRate	double	player's winning bluff rate in past games
winRate	double	player's winning rate
bluffPerGame	integer	bluff number that player makes per game
winNumber	integer	how many game has player won

4.1.1.8.Server Data Entity

Server data entity keeps all the information about players and game tables running on the server module.

Field Name	Data Type	Description
gameTables	GameTable array	list of running game tables on the system
onlinePlayers	Player array	list of online players in the system

4.1.1.9.Macros Data Entity

This entity is really not a data entity, instead , this is an enumarator that keep the information with pre defined unique integers.

Macro Name	Data Type	Description
ON_FLOOR	integer	state info refers that card is on the floor
NOT_DEALED	integer	state info refers that card isn't dealed yet
WAITING	integer	game table is waiting
EMPTY_TABLE	integer	game table is empty
FULL_TABLE	integer	game table is full
RUNNING_GAME	integer	game state that refers game is running
AI_ID	integer	player session id of the AI bot

**All of the macros in the Macro entity has to be unique integers in order not to cause any confussion.*

4.1.2. Client Application Data Entities

Same Data entities can be used in this section.

4.1.3. Database Data Tables

This section explains the database tables of the project. Necessary fields are showed in the tables.

4.1.3.1. User Table

This table keeps the information about users.

Field Name	Data Type	Description
userID	integer	unique id of the user
email	string	email address of the user
username	string	username of the user
password	string	md5 hash of the password
totalPoints	integer	total points of the user

4.1.3.2.Statistics Table

Statistics table keeps the information of the user in the database.

Field Name	Data Type	Description
userID	integer	statistics of which user
bluffRate	float	user's bluff rate in past games
successfullBluffRate	float	user's winning bluff rate in past games
winRate	float	users's winning rate
bluffPerGame	integer	bluff number that user makes per game
winNumber	integer	how many game has player won

4.1.3.3. Rankings Table

Rankings table holds the information of the player's total points.

Field Name	Data Type	Description
userID	integer	reference to User table userID
totaPoints	integer	refenrence to User table totalPoints
winNumber	integer	number of games user won
loseNumber	integer	number of games user lost
winRate	float	winNumber/loseNumber rate

4.2. Data Dictionary

Data	Type	Refer to
AI Agent	sub-module	5.2.1.5.Artificial Intelligence Agent
AI_ID	integer	4.1.1.9.Macros Data Entity
belongs	integer	4.1.1.2.Card Data Entity
bluffPerGame	integer	4.1.1.7.Statistics Data Entity
bluffPerGame	integer	4.1.3.2.Statistics Table
bluffRate	double	4.1.1.7.Statistics Data Entity
bluffRate	float	4.1.3.2.Statistics Table
Card	class	4.1.1.2.Card Data Entity
cards	Card array	4.1.1.3.Player Data Entity
Client Application	module	5.2.2.Android Client Application
CommunicationHandler	sub-module	5.2.1.4.Communication Handler
Database	module	5.2.3.Database
email	string	4.1.1.1.User Data Entity
EMPTY_TABLE	integer	4.1.1.9.Macros Data Entity
FULL_TABLE	integer	4.1.1.9.Macros Data Entity
GameEngine	sub-module	5.2.2.3.Game Engine
GameHandler	sub-module	5.2.1.2.Game Handler
GameMove	class	4.1.1.6.GameMove Data Entity
GameState	class	4.1.1.5.GameState Data Entity
GameTable	class	4.1.1.4.GameTable Data Entity
gameTables	GameTable array	4.1.1.8.Server Data Entity
kind	integer	4.1.1.2.Card Data Entity
LoginHandler	sub-module	5.2.1.1.Login Handler

Macros	class	4.1.1.9.Macros Data Entity
Message	class	7.1.4.Communication Handler
NOT_DEALED	integer	4.1.1.9.Macros Data Entity
Notification	class	7.1.2.Game Handler
number	integer	4.1.1.2.Card Data Entity
ON_FLOOR	integer	4.1.1.9.Macros Data Entity
onFloorCards	Card array	4.1.1.4.GameTable Data Entity
onlinePlayers	Player array	4.1.1.8.Server Data Entity
password	string	4.1.1.1.User Data Entity
Player	class	4.1.1.3.Player Data Entity
players	Player array	4.1.1.4.GameTable Data Entity
PointCalculator	class	7.1.2.Game Handler
Rankings	database table	4.1.3.3.Rankings Table
RUNNING_GAME	integer	4.1.1.9.Macros Data Entity
Server	class	4.1.1.8.Server Data Entity
Server	module	5.2.1.Server
sessionID	long	4.1.1.3.Player Data Entity
state	GameState	4.1.1.4.GameTable Data Entity
Statistics	class	4.1.1.7.Statistics Data Entity
StatisticsHandler	sub-module	5.2.1.3.Statistics Handler
successfullBluffRate	double	4.1.1.7.Statistics Data Entity
successfullBluffRate	float	4.1.3.2.Statistics Table
tableID	integer	4.1.1.4.GameTable Data Entity
tableName	string	4.1.1.4.GameTable Data Entity
turn	integer	4.1.1.5.GameState Data Entity
User	class	4.1.1.1.User Data Entity
User Table	database table	4.1.3.1.User Table
userID	integer	4.1.1.1.User Data Entity
username	string	4.1.1.1.User Data Entity
WAITING	integer	4.1.1.9.Macros Data Entity
winNumber	integer	4.1.1.7.Statistics Data Entity
winNumber	integer	4.1.3.2.Statistics Table
winRate	double	4.1.1.7.Statistics Data Entity
winRate	float	4.1.3.2.Statistics Table

5. System Architecture

A description of the program architecture is presented here.

5.1. Architectural Design

In this project, there are 3 major parts that are server, Android client application and the database.

Server manages:

- ✓ Communication between players
- ✓ Recording statistics into the database
- ✓ Login and logout operations
- ✓ Running the AI agent when necessary

Android Client Application:

- ✓ Provides the cute user interface
- ✓ Running the game
- ✓ Gets inputs from user via the interface

Database:

- ✓ Keeps users' all information
- ✓ Keeps statistics about each user for helping the AI agent
- ✓ Holds ranking tables

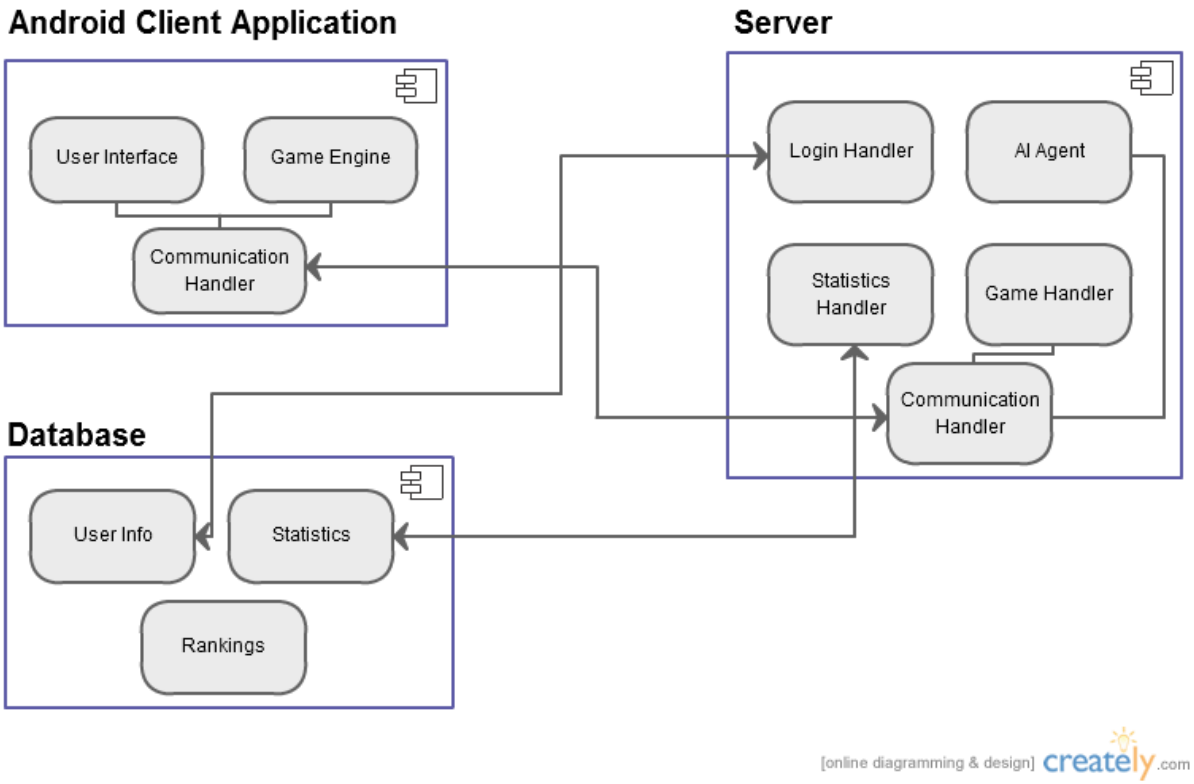


Figure 5.1.a: Component Diagram of the System

When a user opens the client application first the login menu [Figure 6.2.a] will be shown connects to the system, server will check user's username and password from the database. If authentication is done, main menu interface [Figure 6.2.b] will be shown to the user. The user can select one of three options which are "Play Online", "Play vs Computer" and "Settings. If the user selects the "Play Online" option, lobby menu [Figure 6.2.c] will be shown. In the lobby, existing game tables are shown, and the user can join one of them. In the "Play vs Computer" option, user can play with the AI agent. In the "Settings" option menu [Figure 6.2.d], user can change his/her information such as password, e-mail, or profile icon.

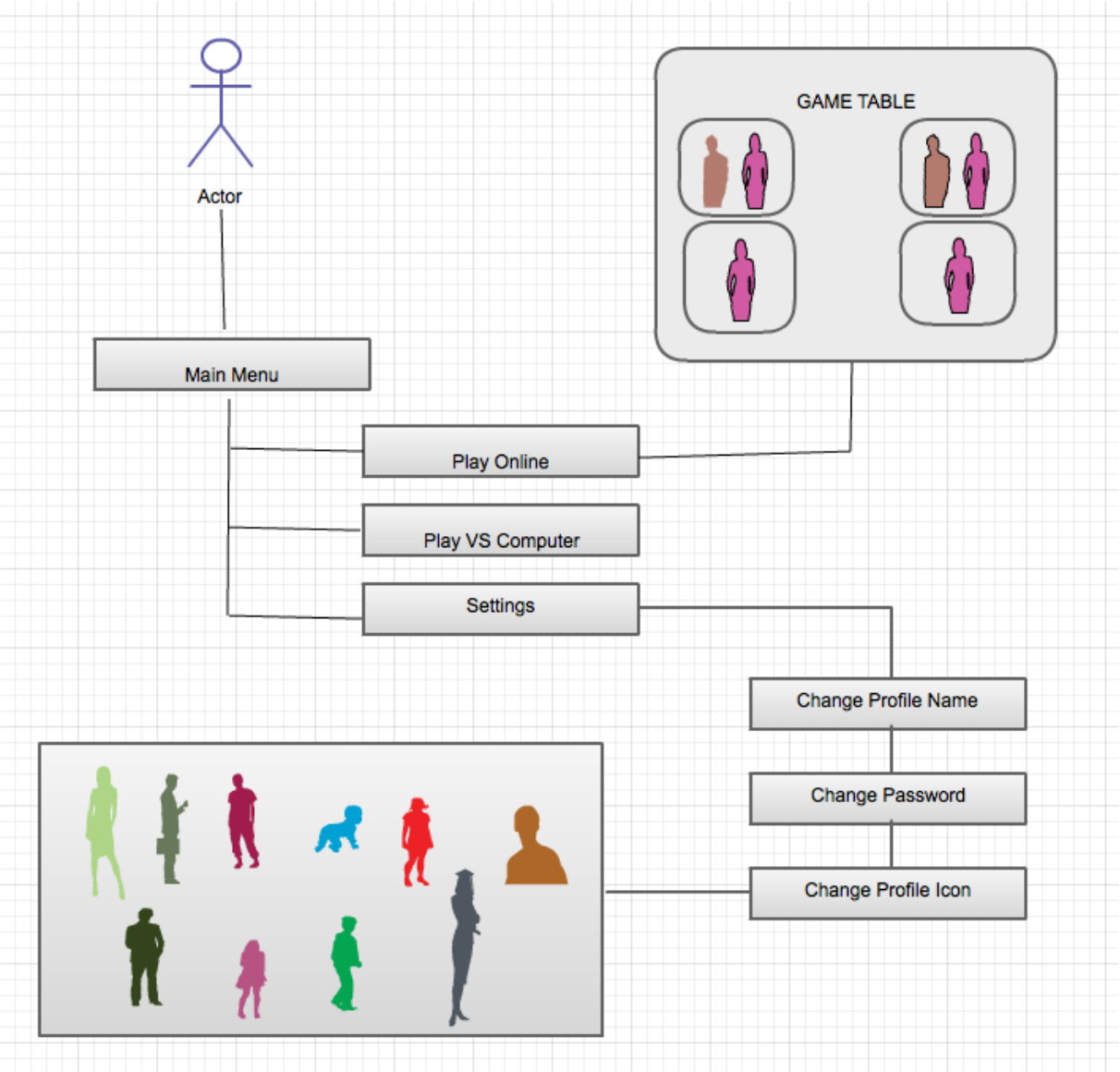


Figure 5.1.b: Use Case of the System

5.2. Description of Components

Employing the object oriented programming paradigm is the best way to ensure the reliability of structured programming techniques, so we are also using object oriented programming paradigm as well in our project. Componentwise structured architecture is very precise and well handled in object oriented systems, the system benefits from this paradigm also from this aspect, because every component should be in maximal consistency among themselves. Modularity is also provided at an extreme level. So it reduces the complexity of project by

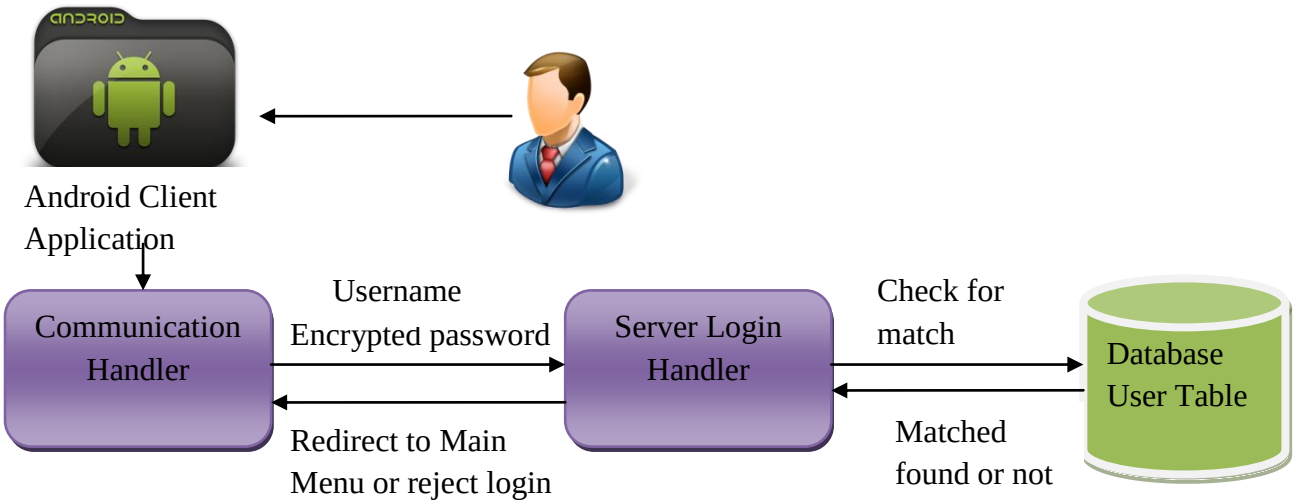
providing ‘part to whole’ approach.Well organized little working parts always increases the success rate of a software project, also this will be convenient to succeed

5.2.1. Server

The server that we used in our project is a PHP server. The reason why we choose PHP for server, is the fact that PHP is the most appropriate script language for server side. Server part has got five components as follows; Login Handler, Game Handler, Statistics Handler, Communication Handler, Artificial Intelligence Handler

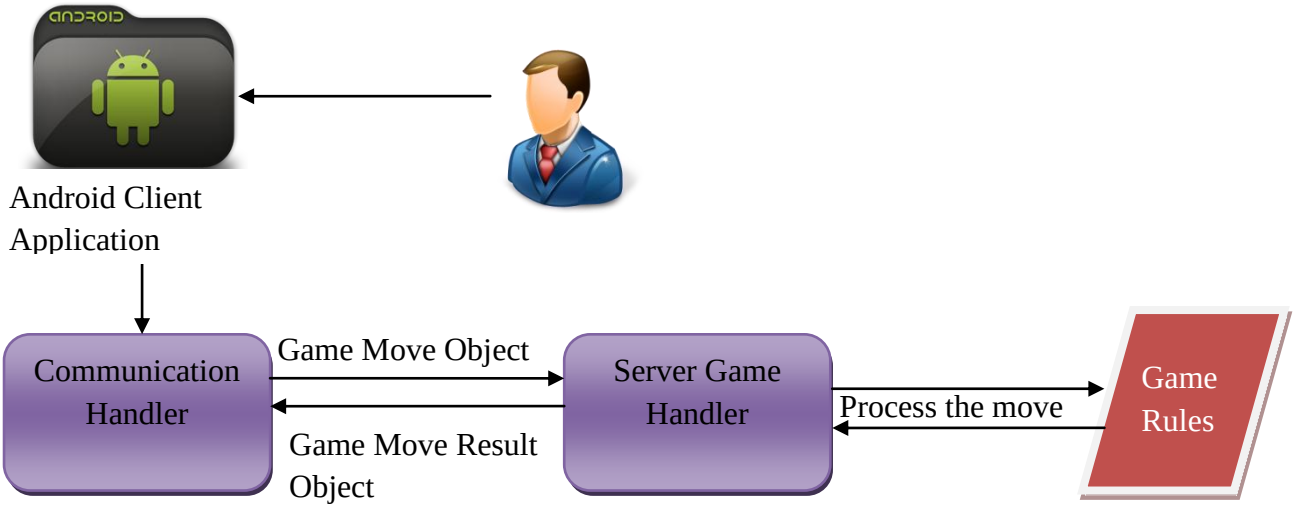
5.2.1.1. Login Handler

In this component of the server module, users’ login attempts will be conducted. Android Client Application will send the username and encrypted password information to the server and Login Handler in the server is about to decrypt the password. Then, username and password will be checked whether they are matched with any entry in the User database. In case of a mismatch, login attempt of user will be rejected. Encryption algorithm will take action when sending the password data online.



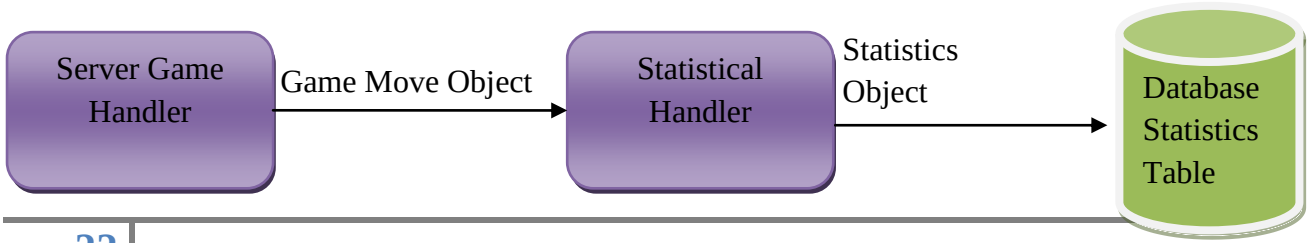
5.2.1.2. Game Handler

In this component of the server module, game rules, moves and other requirements of the game such as scoring, ranking, matching the player with game room, online gamer tables and in-game logs will be handled with this component. Drawing of the cards to players is also done in this component. Move logs of player in game is also recorded and managed within game handler.



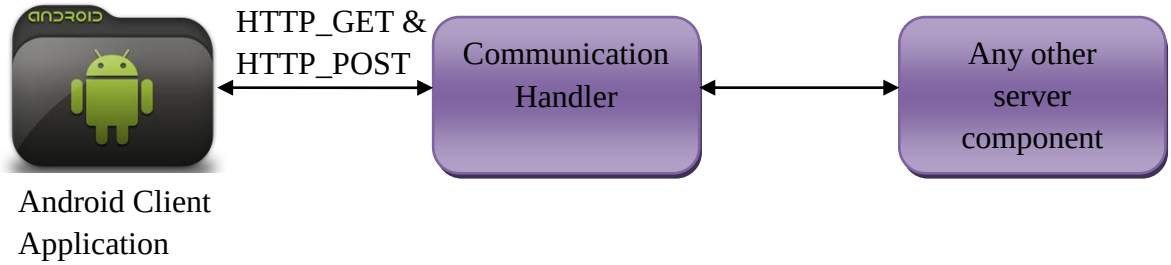
5.2.1.3. Statictics Handler

This module of server is to manage game move objects which comes from the game handler. Game handler sends the items demanded from the statistics handler. Statistics of a corresponding each registered user of game database will be managed and ordered from statistics handler. Move statistics, general rankings, inclinations of a specified gamer will also handled and transmitted to database via this module.



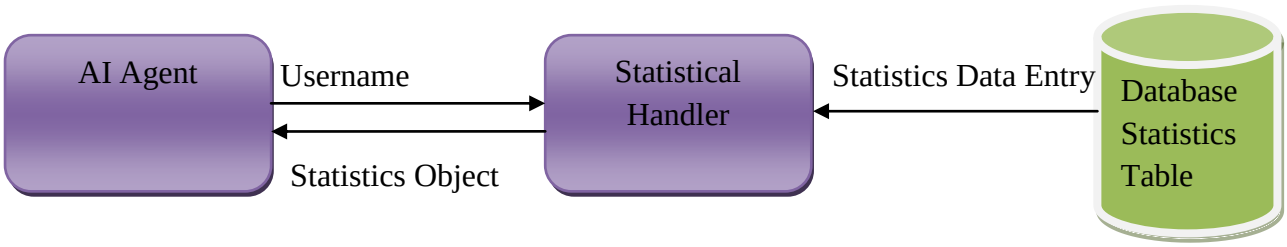
5.2.1.4. Communication Handler

This component handles all the data exchange between the Android Client Application and the server on HTTP port with the help of HTTP_GET and HTTP_POST methods. Game Move Objects will be sent within this handler to server, every data to be sent is transmitted by communication handler.



5.2.1.5. Artificial Intelligence Agent

In this component of the server module, computers countermoves against users moves are determined, game rules and statistics are blended to form a usable game data object. AI agents are administrated via this module. Statistics may affect the behavior of AI agents, for example whether a move is bluff or not will be changed after a series period of games. Learning ability will make the remarkable difference in playing styles. This means that AI agents can change its characteristics according to opponent player.



5.2.2. Android Client Application

This module of the project is client game application running on Android based systems mainly smart phones. ACA will have :

- Nice graphical user interface
- Communication handler that interact with server
- Game engine that runs the game in application

5.2.2.1. Graphical User Interface

Android has very high potential in user interface designs. We are going to use this opportunity in our project. Escpecially the Touch Screen is very useful for games. Drag and drop feature, dynamic menus are also very beneficial points of Android systems.

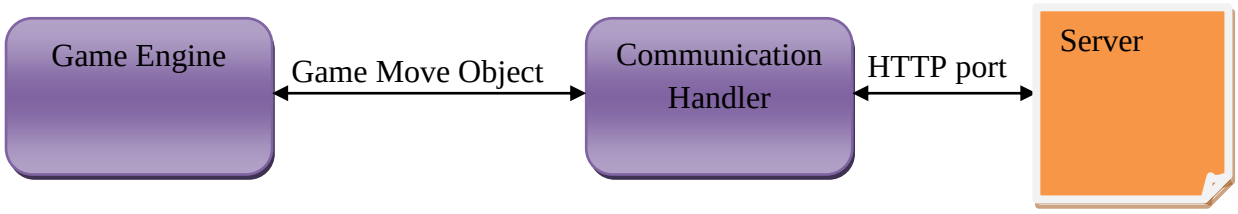
Our ACA has mainly 5 user interface window, we may provide some extra windows later.

- Login Menu window [Figure 6.1.a]
- Main Menu window [Figure 6.1.b]
- Lobby Menu [Figure 6.1.c]
- Settings Menu [Figure 6.1.d]
- In-Game Menu [Figure 6.1.e]

Details of these menus will be explained in 6.User Interface Design.

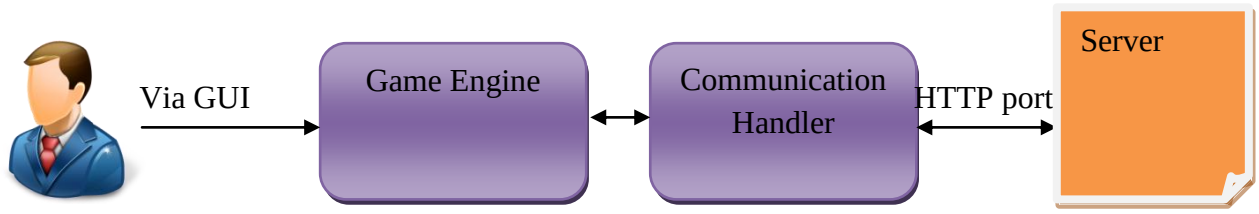
5.2.2.2. Communication Handler

This part of ACA is going to handle all the data interaction with the web server via HTTP port. Every data object that game engine produced will be delivered to the server via Communication Handler. There is no way to interact with outside world for ACA.



5.2.2.3. Game Engine

Game engine handles all the instructions comes from the user via the graphical user interface. Inputs from the gui is converted to input objects and sent to the game engine. Then game engine process the input and produce a proper output. Procuduced output is sent to communication handler and then to the server.



5.2.3. Database

In this project, MySql database will be used and for managing database entries MySql Workbench 5.2.27 is suitable. In this section, details of the database table will not be shown because in section 4.1.3. Database Data Tables those details are explained.

5.3. Design Rationale

We have initially divided the whole system into two modules namely Client Application and Server. Then we realize that, AI Agent needs some database to improve its moves based on past game patterns. Now the modules available in the system are Client Application Server and the Database modules that they are able to work independently from each other.

5.4. Traceability of requirements

In the Software Requirements Specification report has some functional requirements that need to be satisfied in this design report. Those requirements are as follows and corresponding design solutions:

- ✓ Log in to the system : 6.1.a. Login Menu
- ✓ Arranging personal information : 6.1.d. Settings
- ✓ Opening a table : 6.1.c. Lobby
- ✓ Joining an opened game : 6.1.c. Lobby
- ✓ Playing the game or choosing an Al agent on the server : 6.1.b. Main Menu
- ✓ See Rankings : 6.1.e. In-Game Menu

We decided to delete Game History component from the system.

6. User Interface Design

In this section, we will describe user interface of our software project.

6.1. Overview of User Interface

There will be 5 main menus associated with our software project in the user interface. These are:

6.1.1. Login Menu:

The user interface of the system will first expect from the user to input his/her password by theprovided Login Menu. If the user has not had an account yet, he/she will be able to create anaccount by this menu also. If the user inputs wrong log in data the menu will warn the userinforming that wrong log in data is inputted. Unless the user input his/her existing account orcreate one, he/she will not be able to see any further information about the product.

6.1.2. Main Menu:

After creating an account or using an existing account, and logging in the main menu will be seen at the user interface. In this menu the user will be able toplay *blöflü pişti*and adjust his/her settings. In this menu, Play Online, Play vs CPU and Settings options will be displayed.

6.1.3. Lobby Menu:

In the Lobby Menu,the user will able to see rooms, user/s who joined them and their user icons. At this Menu, user can pass to In Game Menu by choosing a room.

6.1.4. Settings:

At the Settings Menu, the user will be able to see profile name, password and choose profile icon options and profile icons. Also, there will be a save button in this menu.

6.1.5. In Game Menu:

At the In Game Menu, the user will able to see his/her and the opponent's profile names and points. Also, game cards in the hand of user and in the table will be displayed at this menu. In addition, there will be an exit button.

6.2. Screen Images

Figure 6.2.a: Login Menu

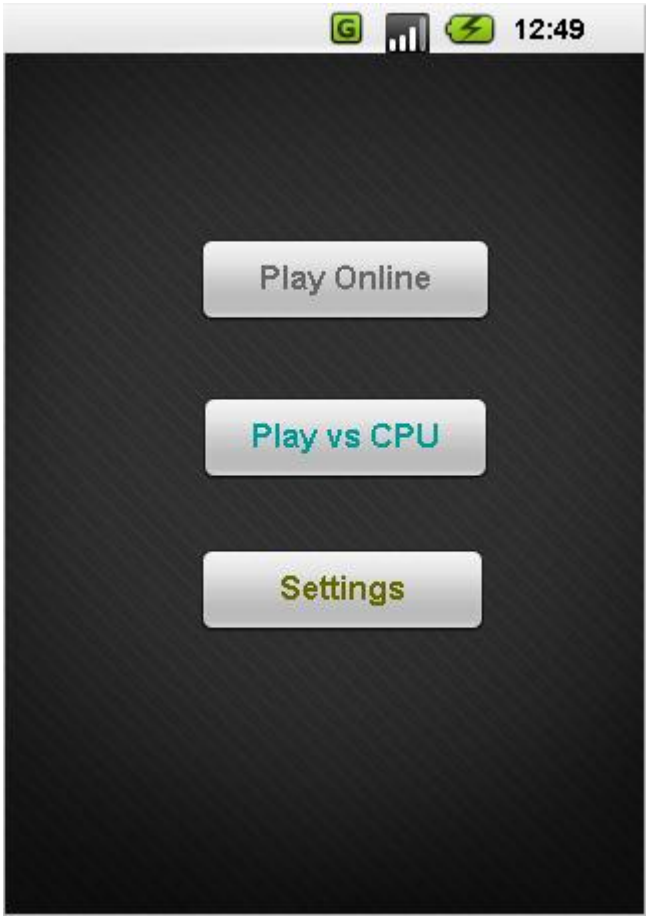
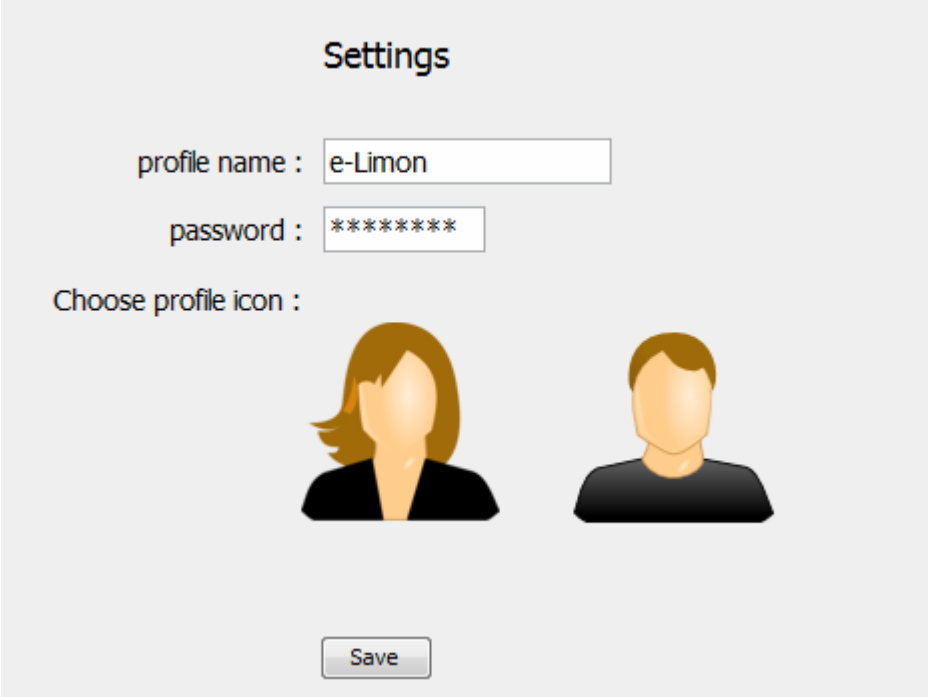


Figure 6.2.b: Main Menu

Figure 6.2.c: Lobby Menu



6.2.d: Settings Menu

6.2.e: In Game Menu

6.3. Screen Objects and Actions

6.3.1. Login Menu

In the user interface, the user will be welcomed by a login menu to the system. In Login Menu, there will be Username, Password, Log In and Sign Up screen objects.

Username and Password: In this spaces, username and password fields will be handled when typed via the virtual keyboard of touch-sensitive device.Behind the background of password box, encryption module will take action, and password

string will not be shown openly, each typed character will be represented as asteriks symbol.

Log In: User will able to pass the Main Menu by this button.

Sign Up: User will able to create an account by this button.

6.3.2. Main Menu

In this menu, there will be Play Online, Play vs CPU and Settings screen objects.

Play Online: User can pass to Lobby Menu by choosing this option.

Play vs CPU: User can pass to In Game Menu by choosing this option.

Settings: User can pass to Settings Menu by chosing this option.

6.3.3. Lobby Menu

In this menu,game rooms will be seen.

Rooms: In this objects, there will be room name, user/s who joined them and their user icons. User can able to join a game by this objects.

6.3.4. Settings

In this menu,profile name, password andprofile iconfields will be seen.

Profile name: In this field, profile name can be typed.

Password: In this field, user can change his/her password and the password characters will be seen as asteriks symbol.

Choose profile icon: User can choose his/her profile icon by the user icons in this field.

Save: User can save his/her information by this button.

6.3.5. In Game Menu

In this menu, there will be profile names, points, game cards and exit objects.

Profile Name: Profile names of the both user will be seen by this item.

Point: Points of the both user will be seen by this item.

Game Cards: Game cards in the hand of user and in the table will be displayed by this items.

Exit: User can leave the room and pass to Lobby Menu by this button.

7.Detailed Design

7.1. Server

7.1.1.Login Handler

Classification: server.LoginHandler

Definition: Deciding mechanism for clients can login or not.

Responsibilities: Login Handler is responsible for processing incoming client login requests. When a client tries to login to the system, the Login Handler will take the username and password information and check with the database and decide to login or reject.

Constraints: Clients send only username and md5 checksum of the password as a string, if somehow those conditions were not satisfied, Login Handler will return an error message to the client.

Composition: There is no sub module under Login Handler.

Uses/Interactions: Client Application's login menu sends a the username and password md5 checksum as a string. Login Handler makes a SQL call to the database and asks for a match for this user.

Resources: Database User table (see: 4.1.3.1 User Table)

Processing: Android client application::Login menu sends 2 string (username,password md5) over HTTP GET method. Those strings takes place at $_GET['username'] and $_GET['password'], then Login Handler makes a SQL call to the database as follows:

```
SELECT * FROM User

WHERE username=$_GET['username'] and
password=$_GET['password']
```

Interface/Exports: If the login is successfull Login Handler sends a notification to the client application : "Login successfull"

Otherwise : "Username or password wrong".

Those notifications will be on a pop-up menu and some time later (appx 1 second) will disappear.

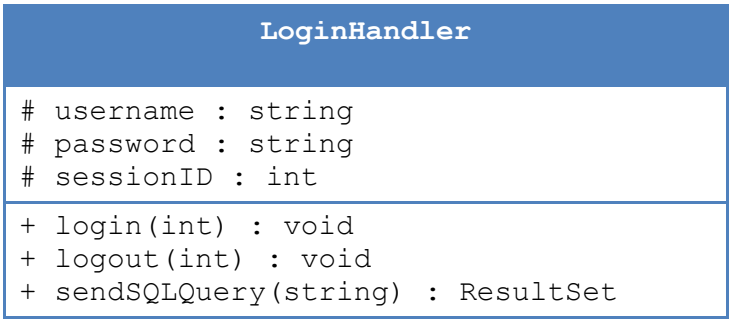


Figure : 7.1.1 Login Handler class diagram

7.1.2.Game Handler

Classification: server.GameHandler

Definition: Game engine that applies game rules.

Responsibilities:

- Dealing cards
- Executing game moves
- Distributing necessary points to players

Constraints: Since the client application game engine directly send data to this module, there will be no wrong structured data.

Composition: Game Handler consists of two parts namely

- **Point calculator**: After the moves have been conducted, it calculates and refreshes the points of the players.
- **Game Move executor**: Takes 1 GameMove object and executes the necessary operations and makes alterations on the datum. There is not a seperate class for this part, GameHandler class is enough for this operation.

Uses/Interactions: GameHandler takes GameMove objects from the CommunicationHandler module and make some process and then send to the it again. When the game is over Game Handler sends necessary data to the Statistics Handler to keep records of how players are playing in terms of bluffing styles.

Resources: GameMove object, GameTable object, Player object.

Processing: Game Handler works in that way:

1. Call 4 times getRandomCard() for putting on the floor
2. Call 8 times getRandomCard() for dealing cards to players
3. Wait for player 1's move
4. Execute move and notify player 2
5. Wait for player 2's move
6. Execute move and notify player 1
7. If Pişti occurs calculate and update points
8. If players do not have any card go back step 2
9. If all cards extinct finish the game and notify players abour the result
10. Update rankings
11. Send statistics to the Statistics Handler

Interface/Exports: This module exports only Notification to the client applications. Notification situations are explained in processing part of this section. Notification can be just an expression or a yes/no question.

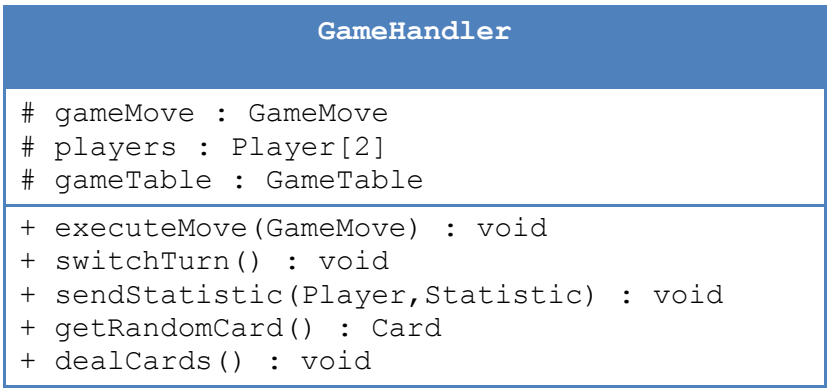


Figure : 7.1.2.a Game Handler class diagram

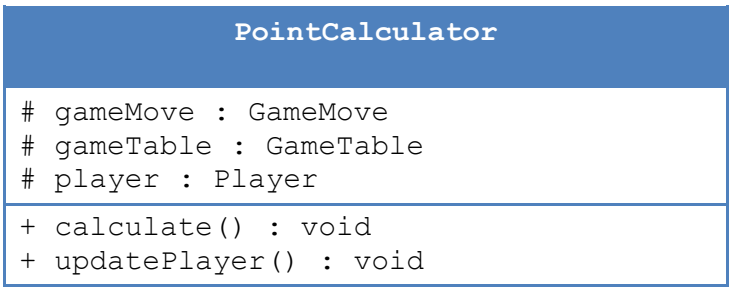


Figure : 7.1.2.b Point Calculator class diagram

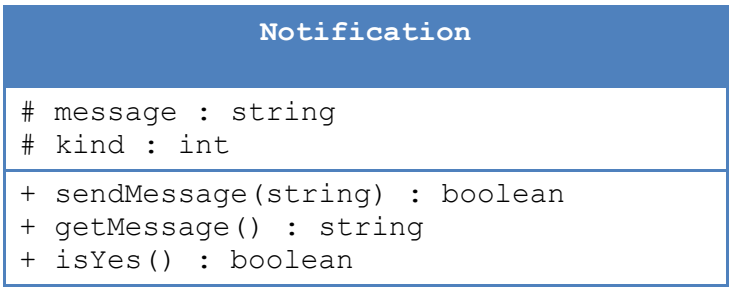


Figure : 7.1.2.c Point Calculator class diagram

7.1.3.Statistics Handler

Classification: server.StatisticsHandler

Definition: A mechanism that handles the user statistics.

Responsibilities:

- Creating Statistics objects
- Sending Statistics objects to Statistic table in the database
- Obtaining statistical data from the database

Constraints: Statistics Handler will be called only each game has over.

Composition: There are 2 parts of this module

- Statistic Saver
- Statistic Getter

Uses/Interactions: AI Agent Handler request a Statistics object from this module. Statistics Handler uses database to get those information.

Resources: Statistic object, User object

Processing: Statistic Saver works in that way :

1. When a game ends, statistics object data will come to the Statistics Handler
2. Makes a SQL call to the database to record the statistic.

Statistic Getter works in that way:

1. Makes a SQL call to the database to get statistic data of a user
2. Gathering these data and then creates a Statistic object and send to the AI Agent .

Interface/Exports: This module creates and sends a Statistics object to AI Agent.

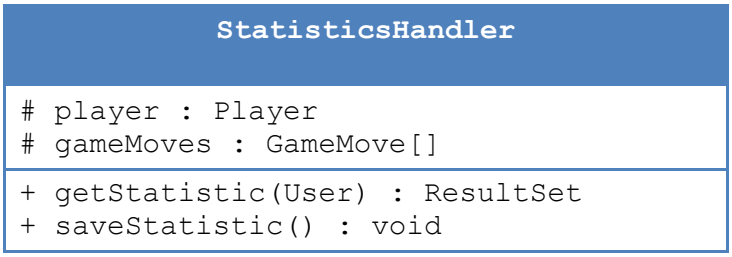


Figure : 7.1.3 Statistics Handler class diagram

7.1.4.Communication Handler

Classification: server.CommunicationHandler

Definition: A mechanism that communicates between server and clients.

Responsibilities:

- Making a connection between server and a client
- Being a bridge between server-clients in terms of data flow over the network.

Constraints: Connection between server and client will be over the HTTP GET and POST methods. There will be no peer2peer connection among clients.

Composition: There is no sub module of Communication Handler.

Uses/Interactions: Since the Communication Handler is a bridge between other server modules and clients, there will be message interface. This interface is explained at Figure 7.1.4.Message class diagram.

Resources: Message interface, server.Server.onlinePlayers

Processing: When a client tries to connect to the server, Communication Handler will recognize its behavior then redirects Login Handler or Game Handler regarding of its session information exists in the server or not. If session information exist that means that client has connected before it can continue to exchange data with the server. Otherwise, that user has not logged in to the system, it has to provide login information to the Login Handler.

Interface/Exports: Communication Handler checks session information wih Server class of the server module. Also the relationship between Login Handler is explained in Processing part.

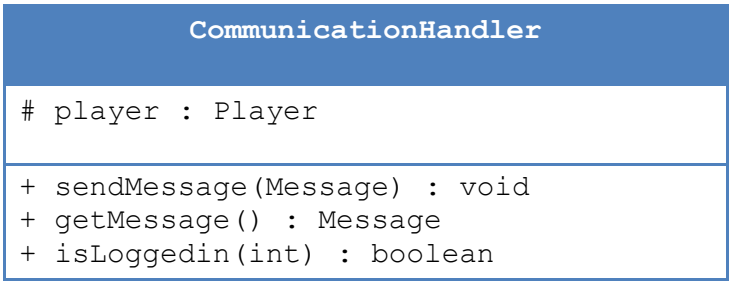


Figure : 7.1.4 CommunicationHandler class diagram

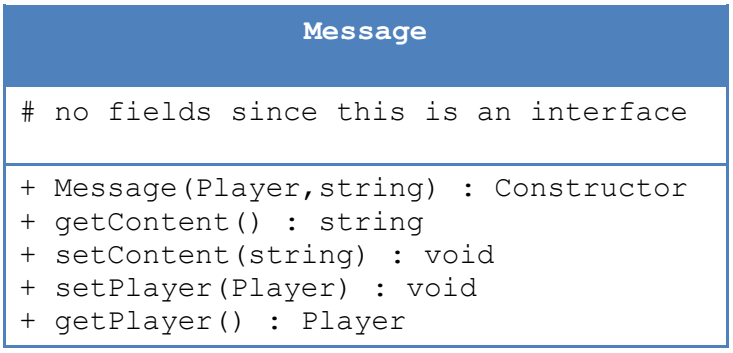


Figure : 7.1.4 Message class diagram

7.2.Android Client Application

7.2.1.Graphical User Interface

Details of this section is explained in 6.User Interface Design.

7.2.2.Commnication Handler

This part is so similar to server.CommunicationHandler

Classification: client.CommunicationHandler

Definition: A mechanism that communicates between server and the client application.

Responsibilities:

- Making a connection between server and the client
- Being a bridge between server-client in terms of data flow over the network.

Constraints: Connection between server and the client will be over the HTTP GET and POST methods. There will be no peer2peer connection among clients.

Composition: There is no sub module of Communication Handler.

Uses/Interactions: Since the Communication Handler is a bridge between other server modules and clients, there will be message interface. This interface is explained at Figure 7.1.4.Message class diagram.

Resources: Message interface, server.LoginHandler

Processing: When the client tries to connect to the server, username and password informatin gathered from Login menu are sent to Communication Handler. Then password is encrypted to md5 checksum and username and password is sent to the server by using HTTP POST method.

Interface/Exports: Server's Communication Handler connects this sub-module.

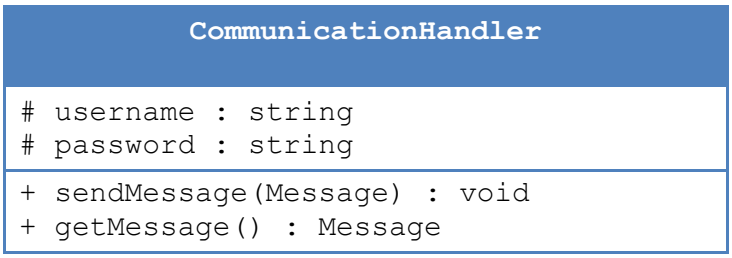


Figure : 7.2.2 CommunicationHandler class diagram

7.2.3.Game Engine

Game Engine is just responsible for getting game move objects from user interface of the client application and deliver to the server via the Communication Handler. All the game running process is executed by server.GameHandler.

7.3.Database

Details of Database module of the project is explained in 4.1.3. Database Data Tables section. There is no further information to mention here.

8. Libraries and Tools

8.1. Server Management Tool

Name of the Software: Xampp Control Panel Application

Version Number: 2.5

Mnemonic: 477 MB

Source: http://www.nat32.com/xampp/xampp25.zip

8.2. Android Development IDE

Name of the Software: Eclipse Java EE IDE for Web Developers

Version Number: Indigo Service Release 1

Mnemonic: 264 MB

Source: http://download.eclipse.org/webtools/downloads

8.3. Database Manager

Name of the Software: MySQL Workbench

Version Number: 5.2.25

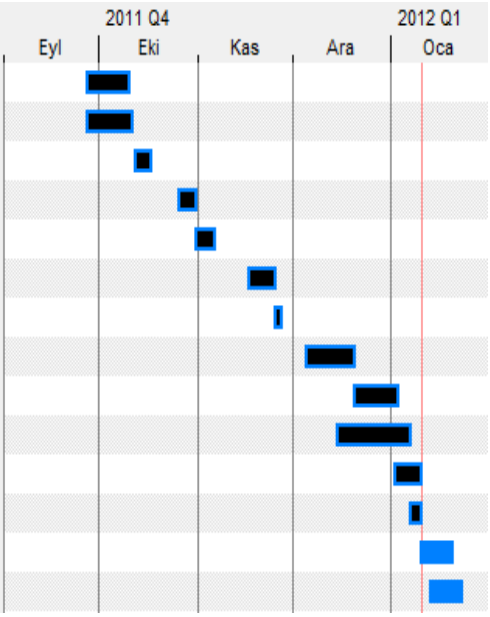
Mnemonic: 57 MB

Source: http://www.mysql.com/product/workbench

9. Time Planning

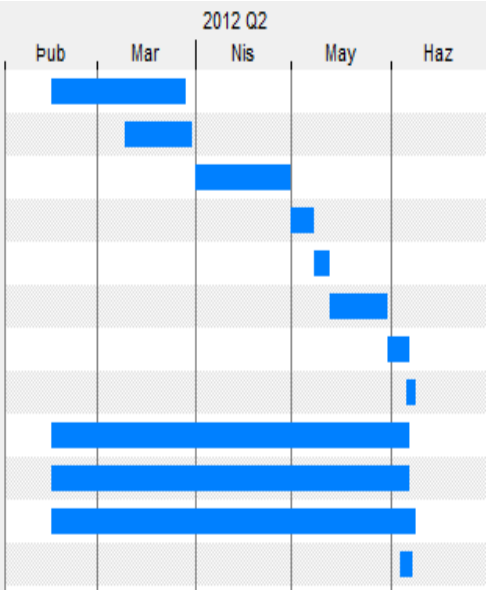
9.1. Term 1

Task Name	Days	Start	Finish	% Done
Project Presentation	14	27.09.11	10.10.11	100
Project Decision	15	27.09.11	11.10.11	100
Pre-proposal	6	12.10.11	17.10.11	100
Proposal	6	26.10.11	31.10.11	100
Market Research	7	31.10.11	06.11.11	100
System Requirement Specifiation	9	17.11.11	25.11.11	100
SRS Review	3	25.11.11	27.11.11	100
Initial Design Report	16	05.12.11	20.12.11	100
Project Presentation Preperation	15	20.12.11	03.01.12	100
Server Implementation	24	15.12.11	07.01.12	100
Detailed Design Report	9	02.01.12	10.01.12	100
Communication Protocol Setup	4	07.01.12	10.01.12	100
Server Client Implementation	11	10.01.12	20.01.12	0
Prototype Demo	11	13.01.12	23.01.12	0



9.2. Term 2

Task Name	Days	Start	Finish	% Done
AI Bot Development	42	16.02.12	28.03.12	0
Graphical User Interface Design	21	10.03.12	30.03.12	0
Integration of Modules	30	01.04.12	30.04.12	0
Integration Testing	7	01.05.12	07.05.12	0
Integration Debugging	5	08.05.12	12.05.12	0
Final Release	18	13.05.12	30.05.12	0
Final Demo	7	31.05.12	06.06.12	0
Project Presentation Preperation	3	06.06.12	08.06.12	0
Code Review	112	16.02.12	06.06.12	0
Testing & Debugging	112	16.02.12	06.06.12	0
Developer Documentation	114	16.02.12	08.06.12	0
User Manuel	4	04.06.12	07.06.12	0



10. Conclusion

This document is prepared to explain the detailed design of the project **Building Server-Client Architecture to Play Card Games, BLÖFLÜ PİŞTİ** supported from e-Limon.
At the earlier steps of the project design, it is implied that what our problem is and what its scope and purpose will be. After that components of the project have been researched. Since our design will be different from others that are in the market, it will be a completely new project, a pathfinder. It is mentioned that what our design should do, what features it will provide, which components it will include, what the user interface will look like and so on. After when we get an initial idea about project, we get into more detail about system architecture and data models. Each module of the system is described. Constraints of the project is discussed. Since this is an initial design document, more details will be provided later in the detailed design report.
Finally, as a plan for the future through the end of the semester, all aspects of the project would be agreed on. It will be a good practice and motivation to make a prototype of project for us when we start the implementation part in the second semester since it will include new topics for all of us such as artificial intelligence and server architecture.