

The Drawing Game

Group 8

Mikael Rydmark

Joel Lasses

Ting-Hey Chau

David Alison

1. Introduction.....	3
1.1 Purpose and Scope of the document.....	3
1.2 The intended audience.....	3
1.3 Version History.....	3
1.4 Related Documents.....	3
1.5 Glossary.....	3
1.6 Summary of the document.....	4
2. System Overview.....	4
2.1 General Description.....	4
2.2 Overall Architecture.....	4
3. Design Considerations.....	9
3.1 Assumptions and Dependencies.....	9
3.2 General Constraints.....	9
4. Graphical User Interface.....	10
4.1 An Overview of the system from a user’s perspective.....	10
4.2 Functional requirements.....	11
4.3 GUI Forms.....	17
5. Design Details.....	19
5.1 Class Responsibility Collaborator (CRC) Cards.....	19
5.2 Class Diagram.....	23
5.3 State Charts.....	24
5.4 Interaction Diagrams.....	24
5.5 Detailed Design.....	26
5.6 Package Diagram.....	35
6. Functional Test Cases.....	37
6.1 General Game.....	37
6.2 Drawing.....	44
6.3 Guessing.....	47

1. Introduction

1.1 Purpose and Scope of the document

The purpose of this document is to give the programmers a clear understanding of what is supposed to be implemented. It is up to the programmers to deliver a product that meets the requirements specified in the Requirements Document for this project. This document will give the programmers some guidance and help in constructing the program.

1.2 The intended audience

The intended audience of this document is primarily the programmers in charge of constructing this project. Other audiences of this document are the other students reviewing this document as well as the project supervisor and any potential customers of this product.

1.3 Version History

Version	Comment	Date	Authors
1.0	First version	2008-03-10	Mikael Rydmark Joel Lasses Ting-Hey Chau David Alison

1.4 Related Documents

Requirement Document 1.0

1.5 Glossary

Client: A program that runs on a personal computer or workstation connected to computer network and requests information from a file server. When a computer is connected to a server, that computer is acting as a client.

Server: A computer that manages centralized data storage or network communications resources. A server provides and organizes access to these resources for other computers linked to it.

Database: A collection of data arranged for ease and speed of search and retrieval by a computer.

DBMS: (Database Management System) a specific server program that handles databases. A couple of well known DBMS are MS-SQL-Server, Oracle, PostgreSQL, Informix, DB2 and MySQL.

VM: (Virtual Machine) is a software implementation of a machine (computer) that executes programs like a real machine. There are two kinds of virtual machines, system virtual machines and process virtual machines. The system virtual machine provides a complete system platform, which supports virtualization of a complete operating system. The process virtual machine is designed to run a single program, which is supporting a single process.

File system: is a method for storing and organizing computer files and the data they contain to make it easy to find and access them

Internet: An interconnected system of networks that connects computers around the world via the TCP/IP protocol.

Java: A Programming language developed by Sun Microsystems that creates code for interactive applications that is executable on web pages by web browsers. These Java applications can execute

on any platform--Mac, PC, etc.

3-Tier: An architecture having three so called tier , consisting of the presentational tier, logic tier and the data tier. Another name for the tier is layer.

GUI: (Graphical User Interface) The use of pictures rather than just words to represent the input and output of a program.

1.6 Summary of the document

This document consists of six parts. The first part is this one, the introduction. The second part is the system overview describing how the system is made up. In the third part we describe on what assumptions the system is based on and what considerations need to be taken into consideration when constructing this system.

The fourth part describes the graphical user interface which is an important part giving you a clearer picture of what is actually constructed. Part five gives us a more detailed view of what different classes and methods the system consists of. Finally the sixth part consists of a complete set of functional test cases for system testers to use. These tests are designed to make sure the system developed actually meets all the requirement specified in the Requirement Document for this project.

2. System Overview

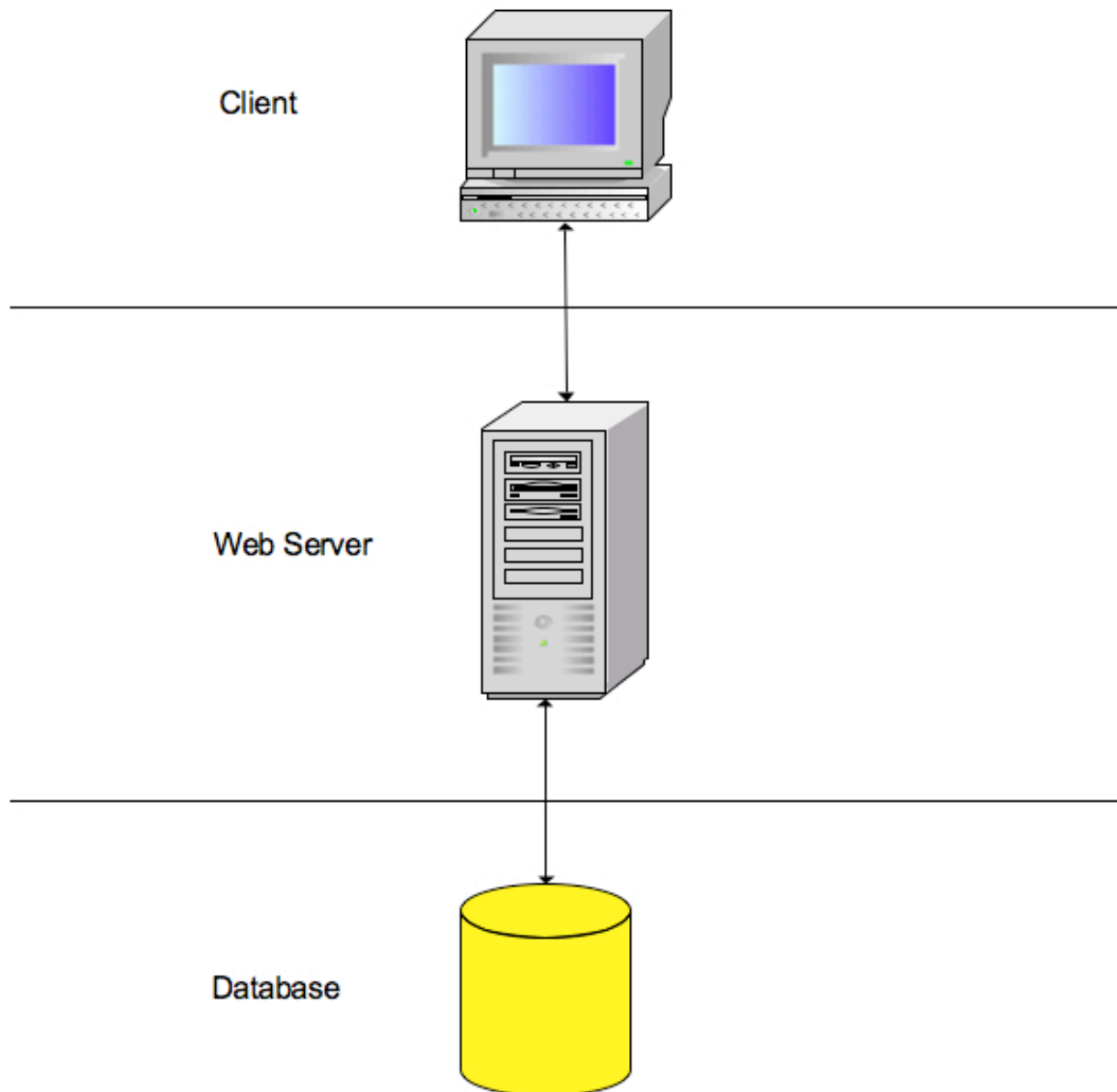
2.1 General Description

The Drawing Game is pretty much what it sounds like. It's a game where you are supposed to draw! The game is supposed to be a multiplayer game played over the web. The game concept is quite simple. There are two teams playing and players take turn drawing a specific word. The team that manages to guess the correct word wins that round and moves a number of squares on the game board. This is the basics of the game (more detailed rules do apply but I will not bore you with them right now).

For this system to work over the web we need a running web server connected to a database and a file system. We also need an applet or similar running for every client connected to the server. The clients do not interact directly with each other but through the server.

2.2 Overall Architecture

The drawing game is a three-tier client-server system as shown in the model below.



The system consists of three major parts, the client, the web server and the database.

The Client

The Client interacts with the game by using a web browser on a computer connected to the Internet. The client can access the game by typing in the address of the game site in the browser. Once relocated to the correct address the client is provided with information from the web server running the game. The client then continues to interact with the game by using his/her keyboard and mouse.

The Web Server

The Web Server is the brain of the whole system and contains all code required to run the game. The server interacts with the clients connected to the game and takes appropriate actions depending on what the users does. If necessary it communicates with the database and file system.

The Database

The Database is a big storage and contains information about registered members. It also holds a

large number of words used by the web server in the game. The words in the database are the ones used by the clients when playing the game, i.e. the words to draw.

2.3 Detailed Architecture

The diagram later in this section shows the control and data flow of the system. The different modules in the diagram are described in detail below.

The Client

Interpret command

The command interpreter listens for actions from the client. When the user clicks a button or something similar the command interpreter examines what the user wants to do and if it is a valid command it sends the command to the server communicator.

Process information

This part of the systems waits for information to be received from the server communicator. When something is received the module processes the information and determines what kind of information it is and displays it to the client. The information can be anything from a new line drawn in the game or user information and high score lists.

Server communication

The Server communicator waits for a command to be received from the command interpreter or information to be sent from the client communicator on the web server side of the system. When the server communicator receives a command from the command interpreter it sends the command to the web server (the client communicator) for execution.

When receiving information from the server side, the server communicator send that information to the clients' information processor.

The Web Server

Client communication

This module communicates with the client side of the system. It receives commands form the client, specifically from the server communicator. When a command has been received the module sends it to the action handler for execution.

When an action has been executed the client communicator receives information about this from the action handler and then goes ahead and sends the information to the server communicator so the information can be displayed to the clients.

Action handler

The action handler is the heart of the system. It receives commands from the user via the client communicator. The action handler then proceeds to execute the command. If necessary it communicates with the database and/or file system via the database/file system communicators. When the command has been executed appropriate information is sent to the clients via the client communicator.

Database communication

Design Document
DD1363 MVK
2008-03-10

The database communicator waits for the action handler to tell it what to do. When a command is received it communicates with the database and retrieve, updates, inserts and/or deletes the necessary information.

File system communication

The file system communicator accesses the file system and sends appropriate information to the action handler.

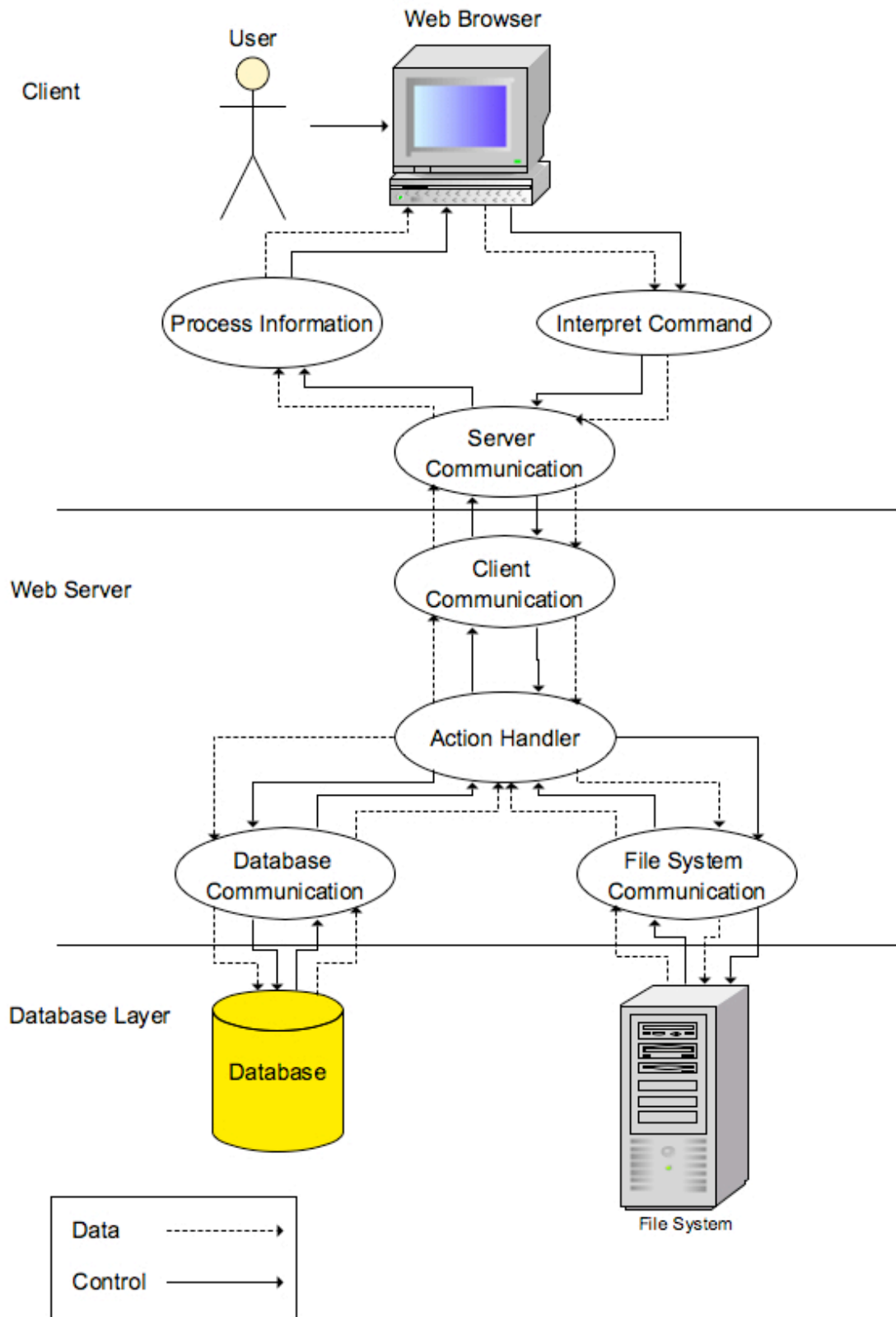
The Database Layer

The Database

This part is described in section 2.2 above.

The File system

The File system contains all other necessary information that is not stored in the database, e.g. profile pictures.



3. Design Considerations

3.1 Assumptions and Dependencies

The server part of the system will run on any platform that supports Java Virtual Machine of version 1.5 or higher. The web client requires a modern web browser that supports java applets. We will assume it is possible to use a SQL database run on one of the school's computers.

We assume that the end users of this game have some computer online game experience and are somewhat familiar with drawing on a computer.

3.2 General Constraints

Some factors affecting the performance of the system is network capacity, web server capacity and database server capacity. The system is also space dependent since we need to store user profiles, high scores and all available words to draw in the game.

4. Graphical User Interface

4.1 An Overview of the system from a user's perspective

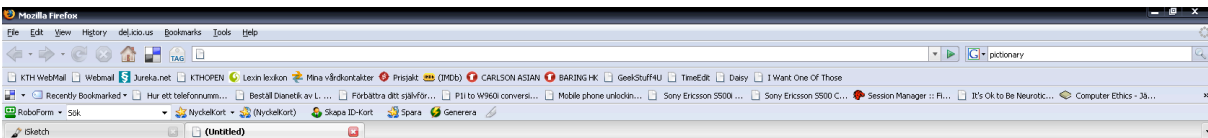
The Drawing Game Client

(Being used in the web browser)

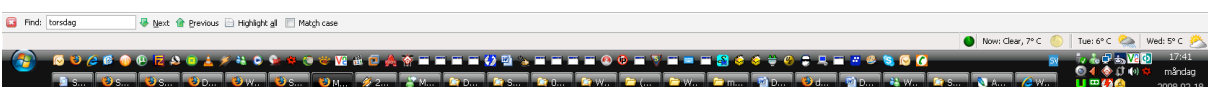
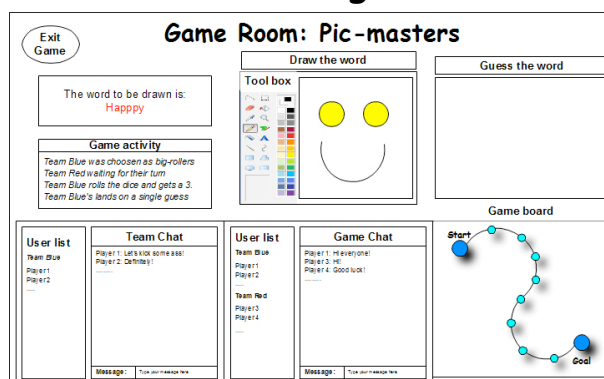
When the user selects “Login“ a new window will appear, given that the username and the password are correct, the main window of the system will appear. If the login fails, a red text indicates that the user has entered an incorrect login name or password. Login is required if the users wants their high-score to be stored and others to be able to view their profile.

While signed onto the main system, the user is able to chat with other players that also are signed in to the system, that still have not joined a specific game room yet. After joining a game room a user can easily leave by clicking on the “Leave Room” or choose “Exit Game” to get back to the main menu.

Editing a profile is easily done by clicking on “Edit Profile” in the main window; by clicking that button a new window will be displayed. Help for playing the game is easily provided by clicking on the “Tutorial” button. An overview picture how the website and the system could look like in the web browser is shown below.



The Drawing Game



Overview over the website

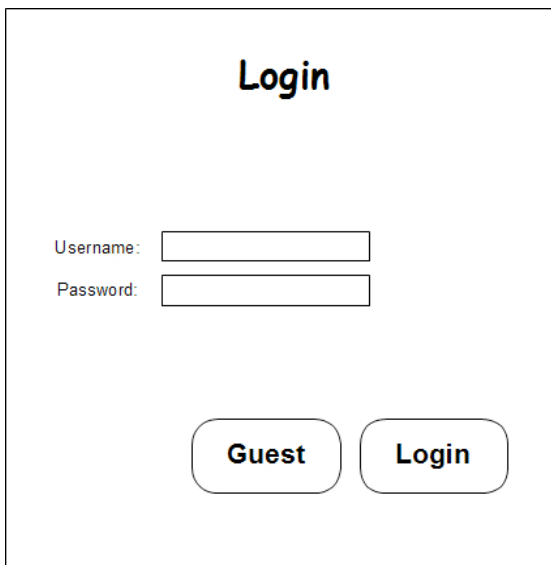
4.2 Functional requirements

Login screen

When starting the program, the main window appears in the user's web browser. The user has two options to choose between to get access to the system, either sign in with an already existing account or sign in as a guest.

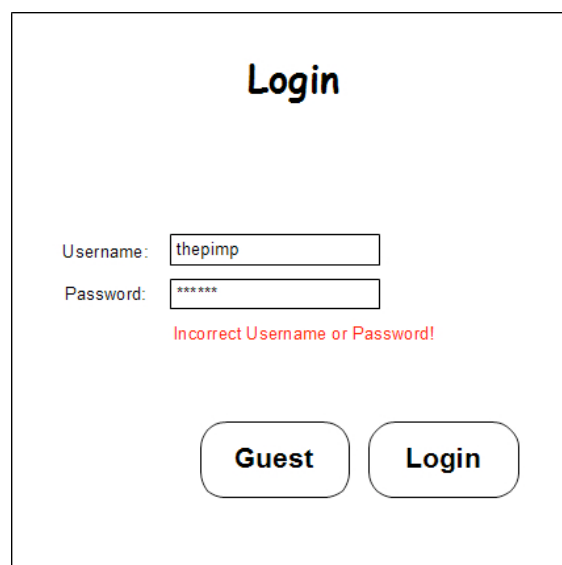
Functional requirements:

Unauthorized users should not be able to login to others accounts.



The image shows a login form titled "Login". It contains two input fields: "Username:" and "Password:". Below the input fields are two buttons: "Guest" and "Login".

Form 1



The image shows a login form titled "Login". The "Username:" field is filled with "thepimp" and the "Password:" field is filled with "*****". Below the input fields, there is a red error message: "Incorrect Username or Password!". Below the error message are two buttons: "Guest" and "Login".

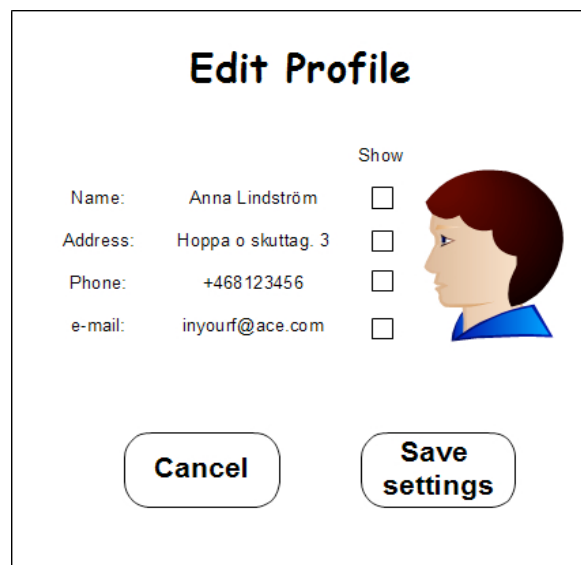
Form 2

Edit Profile

Displays already registered profiles and enables the user to edit their information, which kind of information they want others to see, for example name, e-mail and much more. By checking on the check boxes besides specific information in the profile, that specific information will be show in the public profile. But not until the user clicks on "Save Settings"-button or "Cancel" to ignore the changes and go back to the main system window.

Functional requirements:

Do not display unchecked information in public
Store changes to the profile.

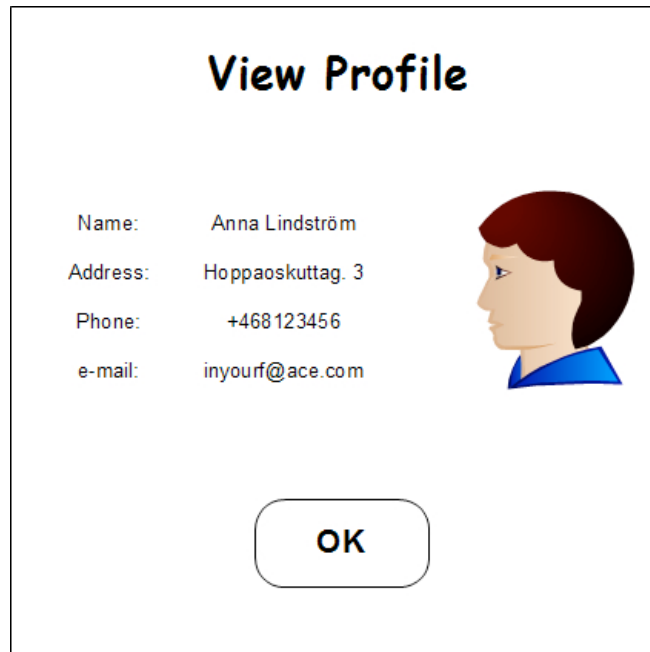


The image shows an "Edit Profile" form. It contains a list of information with checkboxes to the right of each item. The information is: Name: Anna Lindström, Address: Hoppa o skutttag. 3, Phone: +468123456, and e-mail: inyourf@ace.com. To the right of the list is a profile picture of a woman. Below the list are two buttons: "Cancel" and "Save settings".

Form 3

View Profile

By clicking on a user in the chat or during a game, that specific users profile will be displayed, given that the user is a registered user. The profile window enables the user to view more information about a registered user. The displayed information depends on the users settings, which is customizable through “Edit Profile” in the main window in the system.



The image shows a dialog box titled "View Profile". It contains the following information:

Name:	Anna Lindström
Address:	Hoppaoskuttag. 3
Phone:	+468123456
e-mail:	inyourf@ace.com

There is a profile picture of a woman with short brown hair and a blue top. At the bottom center of the dialog box is a rounded rectangular button labeled "OK".

Form 4

Main window

This includes the tutorial, viewing of profiles, chatting with other players, joining and creating game rooms.

The Drawing Game

Game rooms list	User list	Chat
Game Room 1 Game Room 2 Game Room 3 Game Room 4	Player 1 Player 2 Player 3 Player 4	Player 1: Hi everyone! Player 3: Hi! Player 4: Good luck!
		Message: <input type="text" value="Type your message here"/>

[Create Room](#) [Join Room](#) [Tutorial](#) [Edit Profile](#)

Form 5

Functional requirements:

- Chat with other users
- Join already existing Game Rooms
- Create a new Game Room
- Watch the tutorial
- Edit profile

Create Game Room

When creating a new game room, one can choose name of the game, which makes it easier for friends and other players to join. The creator can also choose how many numbers of players he/she wants in the game. He/she also has the option to choose a password, which makes the game room private and leaves unwanted players out.



The image shows a dialog box titled "Create Game Room". It contains three input fields: "Name:" with a text box containing "Your preferred game name", "Number of players:" with a dropdown menu showing "1", and "Password (optional):" with a text box containing "Your preferred password". Below the input fields are two buttons: "Cancel" and "Create Room".

Form 6

Functional requirements:

- Add name to the new game room
- Choose number of players
- Add a password (optional)
- Create the new game room

View a Game room

You can only watch a game room as long as it is still active and waiting for players. Only the game room creator can choose "Start Game" to start the game. Unless the minimum numbers of players are 4 (including the creator), the "Start Game" button is disabled. The game room chat is located in the middle of the window. The user has the option to choose between the two different teams by clicking on the "Join" button to jump directly to a specific team. If the user picks the wrong team, clicking on the "Join" button located on the opposing teams "Join" -button can easily change it. By clicking on the "Leave Room" button one can always go back to the main system window and choose another game room.

Game Room: Pic-masters

Game Room Chat

User list

Team Blue

Player 1

Player 2

Team Red

Player 3

Player 4

Player 1: Hi everyone!

Player 3: Hi!

Player 2: Finally 4 players! =>

Player 4: Good luck!

Message:

Form 7

Functional requirements:

Chat with other players

Change between the two teams

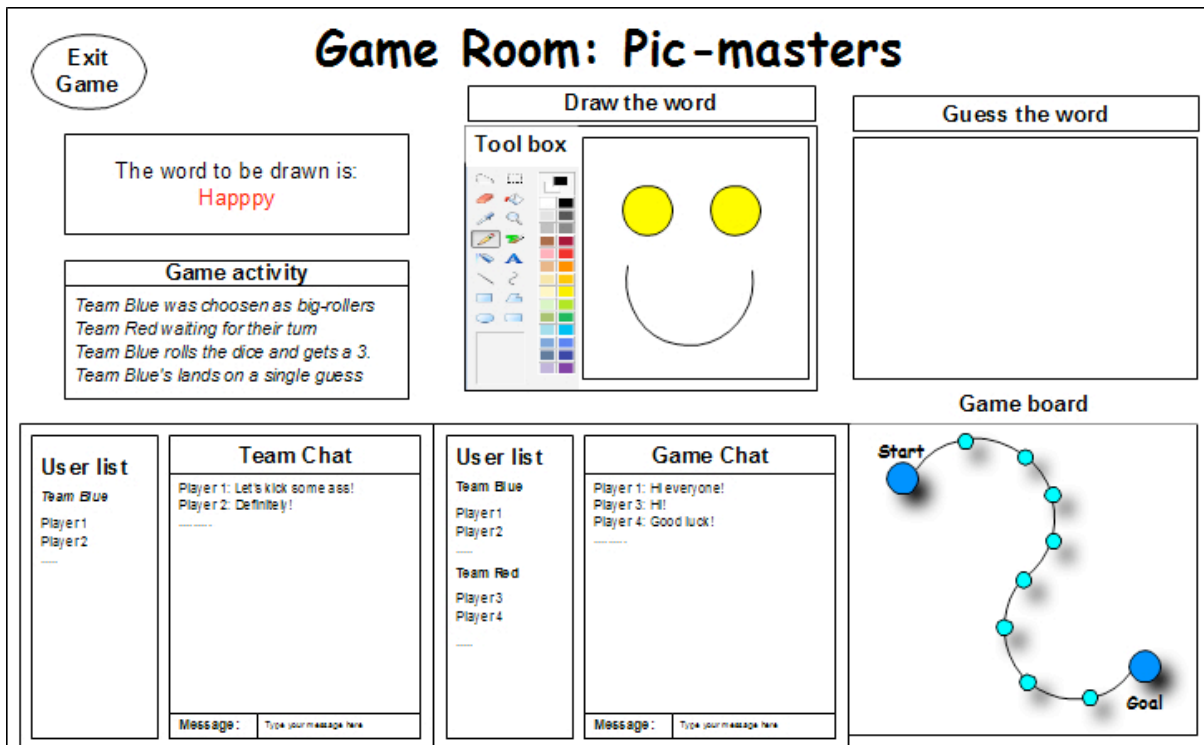
The game creator should be able to start the game

Be able to leave the game room

Playing the game

The game board is available in the lower right section, while the team chat and the game chat is located on the bottom left. The drawing section with an easy to use toolbox is located in the middle of the game window. At the top right window the current drawn word is shown.

Depending on which role the user has during a game, some parts of the game windows is faded and unavailable. For example when it is the users turn to draw, both the game chat and the team chat will be disabled. The drawing section in the middle is faded when it is time to guess a word that one's teammate is drawing.



Form 8

Functional requirements:

- Be able to chat with other players
- Be able to guess on a drawing
- Be able to draw when it is time to draw
- Be able to exit

4.3 GUI Forms

Form 1

Names: The form has a login field called “Login” and “Password”.
The form also has two buttons, called “Guest” and “Login”

Triggered by: The user clicks on the “login” button on the first window

Controls

Login Button: A login attempt is made with the data from “Login” and “Password”.

Form 2

Names: The same as form 1

Triggered by: The data from “Username” or “Password” was incorrect

Controls

Login Button: A login attempt is made with the data from “Username” and “Password”.

Form 3

Names: Checkboxes for showing information in public called “nameBox”, “addressBox”, “phoneBox” and “emailBox”.
The form also has buttons called “Cancel” and “Save Settings”

Triggered by: The user clicks on the “Edit Profile” from the main window (form 5)

Controls

Save settings Button: updateProfile()
Cancel Button: Return back to the previous window

Form 4

Names: The form has one button “OK”

Triggered by: The user clicked on someone’s name in the chat

Controls

OK Button: Returns the user to the main window

Form 5

Names: The form has one field called “mainChat”.
The form also has four buttons “Create Room”, “Join Room”, “Tutorial” and “Edit Profile”

Triggered by: The user signed in either as a guest or as a registered user

Controls

Create Room Button: createGameRoom(), form 6 displayed

Join Room Button: `joinGameRoom(nameoftheroom)`, form 7 displayed
Tutorial Button: `tutorial()`
Edit Profile Button: `editprofile()`

Form 6

Names: The form has three fields, name of the Game Room which is named "gameRoomName", numbers of players slots available in the game room "numberOfPlayers" and passwordfield which is optional, called "gameRoomPassword".
The form has also two buttons "Cancel", "Create Room"

Triggered by: The user clicked on "Create Room" in the main window (form 5)

Controls

Create Room Button: `createGameRoom(name_of_choice)`
Cancel Button: Return back to the previous window

Form 7

Names: The form has four fields, name of the Game Room which is named "gameRoomName", numbers of players slots available in the game room "numberOfPlayers" and passwordfield which is optional, called "gameRoomPassword" and "gameRoomChat".
The form has also four buttons "Leave Room", "Start Game" and two "Join" buttons.

Triggered by: The user clicked on "Create Room" in the window Create Game Room (form 6)

Controls

Start Game Button: `startGameRoom(name_of_the_game_room)`, form 8 displayed
Leave Room Button: Return back to the previous window (main window form 5)
Join Button: `joinTeam1()` or `joinTeam2()` depending on which one is being clicked.

Form 8

Names: The form has two fields, one field is called "gameChat", while the other one is called "teamChat".
The form has three additional graphical areas, one which shows the game board, the second one showing the drawing section and the third one showing the drawing that the drawer has drawn/is drawing.
The form has also one button "Leave Game"

Triggered by: The user clicked on "Start Game" in the window Create Game Room (form 7)

Controls

Exit Game Button: Return back to the main window (form 5)

5. Design Details

5.1 Class Responsibility Collaborator (CRC) Cards

LoginFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the login frame	<u>MainFrame</u>

MainFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the main frame	<u>LoginFrame</u> <u>ProfileFrame</u> <u>GamesettingsFrame</u>

StartgameFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the start game frame	<u>GamesettingsFrame</u>

GamesettingsFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the game settings frame	<u>StartgameFrame</u>

GameFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the game frame	<u>StartgameFrame</u> <u>MainFrame</u>

ProfileFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the profile frame	<u>MainFrame</u>

DbConnection	
<i>Responsibilities</i>	<i>Collaborators</i>
Connect to the database	<u>LoginFrame</u> <u>GameFrame</u>

TheGame	
<i>Responsibilities</i>	<i>Collaborators</i>
A game from beginning to the end	<u>GameFrame</u>

Player	
<i>Responsibilities</i>	<i>Collaborators</i>
Information about a player	<u>GameFrame</u> <u>StartgameFrame</u>

Server	
<i>Responsibilities</i>	<i>Collaborators</i>
Handles the communications	<u>ServerCommunication</u>

ServerCommunication	
<i>Responsibilities</i>	<i>Collaborators</i>
Communicates with the server	<u>Server</u> <u>Chat</u>

Chat	
<i>Responsibilities</i>	<i>Collaborators</i>
Sends and receives text string to and from the server	<u>MainFrame</u> <u>ServerCommunication</u> <u>GameFrame</u>

Canvas	
<i>Responsibilities</i>	<i>Collaborators</i>
A drawing area for the player	<u>ServerCommunication</u> <u>GameFrame</u>

ToolbarFrame	
<i>Responsibilities</i>	<i>Collaborators</i>
GUI for the toolbar	<u>Line</u> <u>Pen</u> <u>Circle</u> <u>Rectangle</u> <u>CircleFilled</u> <u>RectangleFilled</u> <u>GameFrame</u> <u>AbstractTool</u>

AbstractTool	
<i>Responsibilities</i>	<i>Collaborators</i>
An abstract class for all painting tools to inherit from	<u>Line</u> <u>Pen</u> <u>Circle</u> <u>Rectangle</u> <u>CircleFilled</u> <u>RectangleFilled</u> <u>ToolbarFrame</u>

Line	
<i>Responsibilities</i>	<i>Collaborators</i>
Draws a line on canvas	<u>AbstractTool</u> <u>ToolbarFrame</u>

Pen	
<i>Responsibilities</i>	<i>Collaborators</i>
Draws with pen on canvas	<u>AbstractTool</u> <u>ToolBarFrame</u>

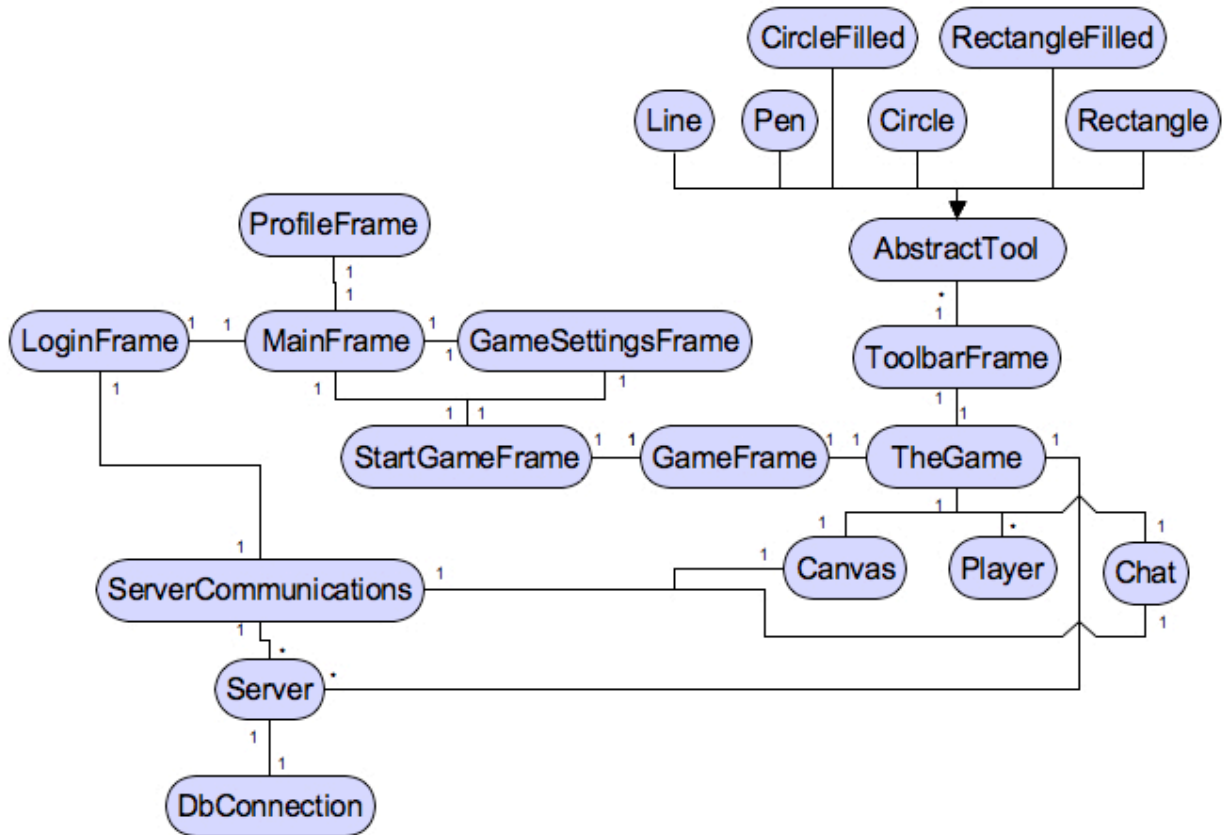
Circle	
<i>Responsibilities</i>	<i>Collaborators</i>
Draws a circle on canvas	<u>AbstractTool</u> <u>ToolBarFrame</u>

CircleFilled	
<i>Responsibilities</i>	<i>Collaborators</i>
Draws a filled circle on canvas	<u>AbstractTool</u> <u>ToolBarFrame</u>

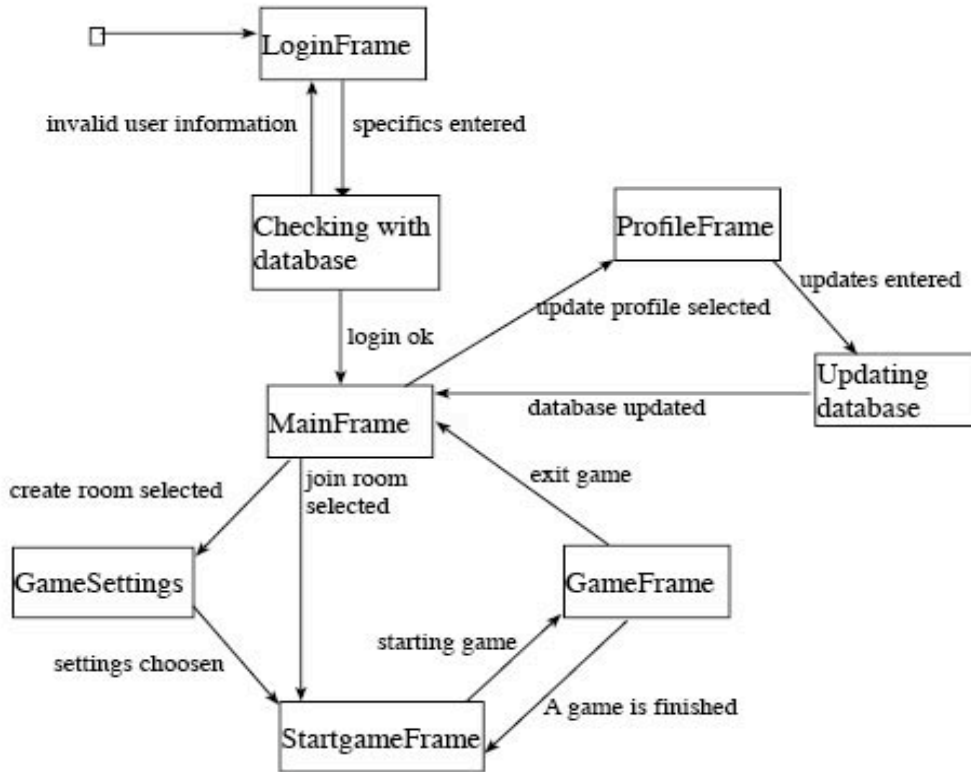
Rectangle	
<i>Responsibilities</i>	<i>Collaborators</i>
Draws a rectangle on canvas	<u>AbstractTool</u> <u>ToolBarFrame</u>

RectangleFilled	
<i>Responsibilities</i>	<i>Collaborators</i>
Draws a filled rectangle on canvas	<u>AbstractTool</u> <u>ToolBarFrame</u>

5.2 Class Diagram

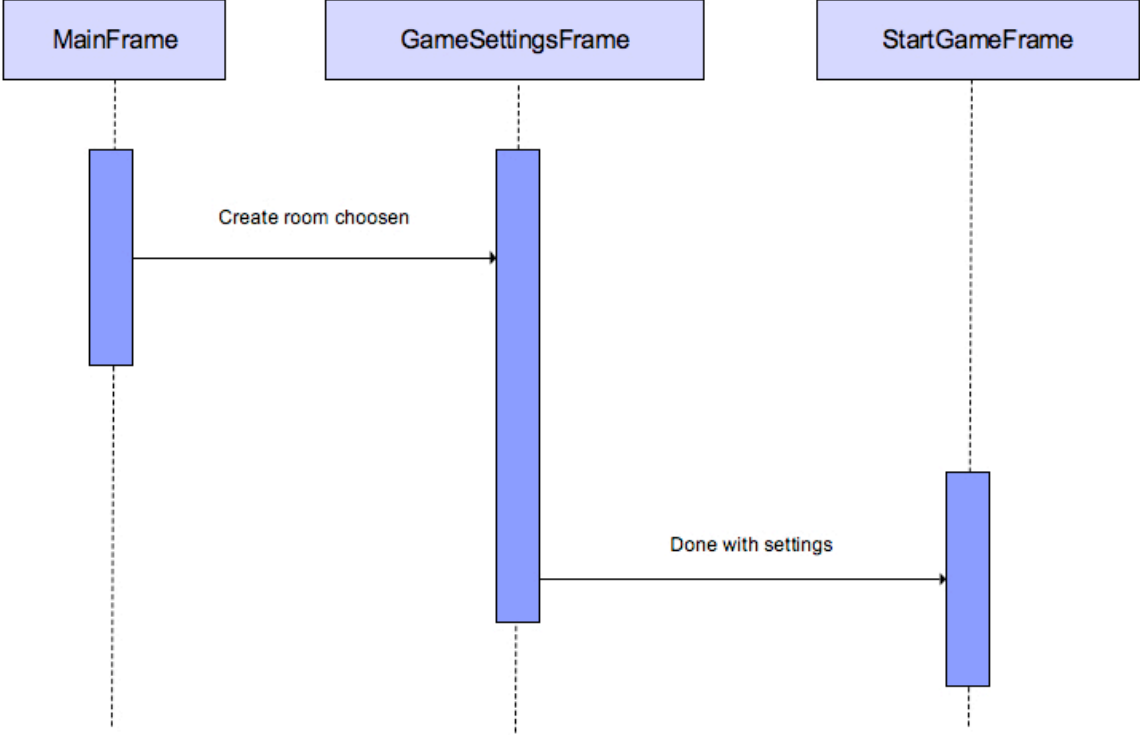


5.3 State Charts

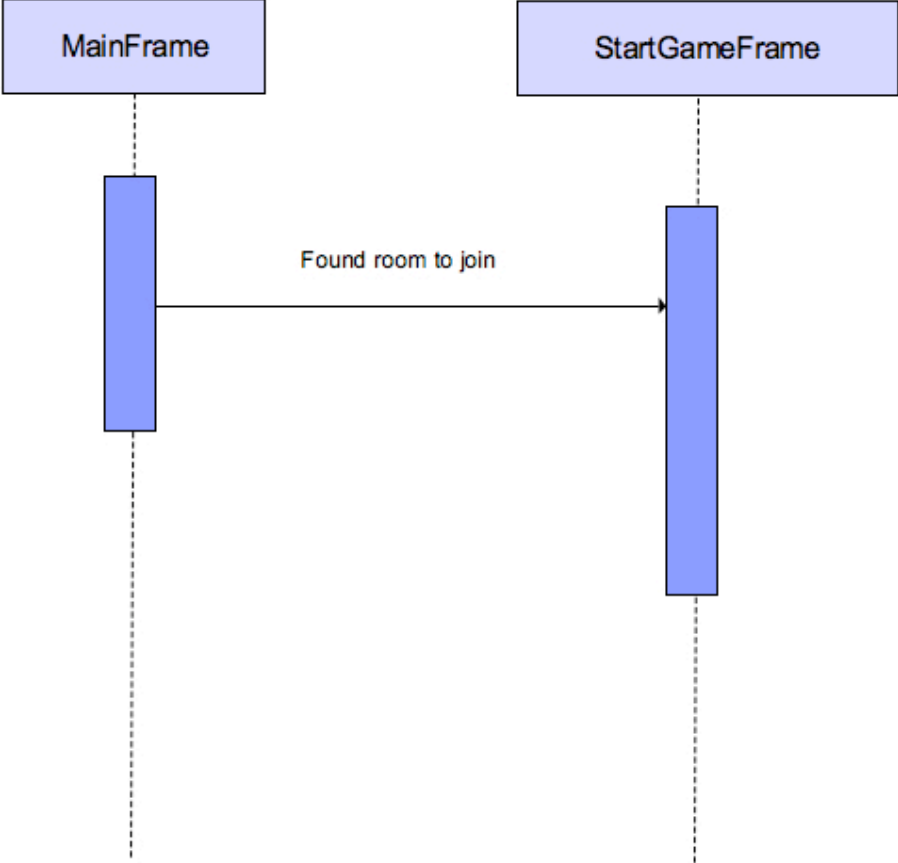


5.4 Interaction Diagrams

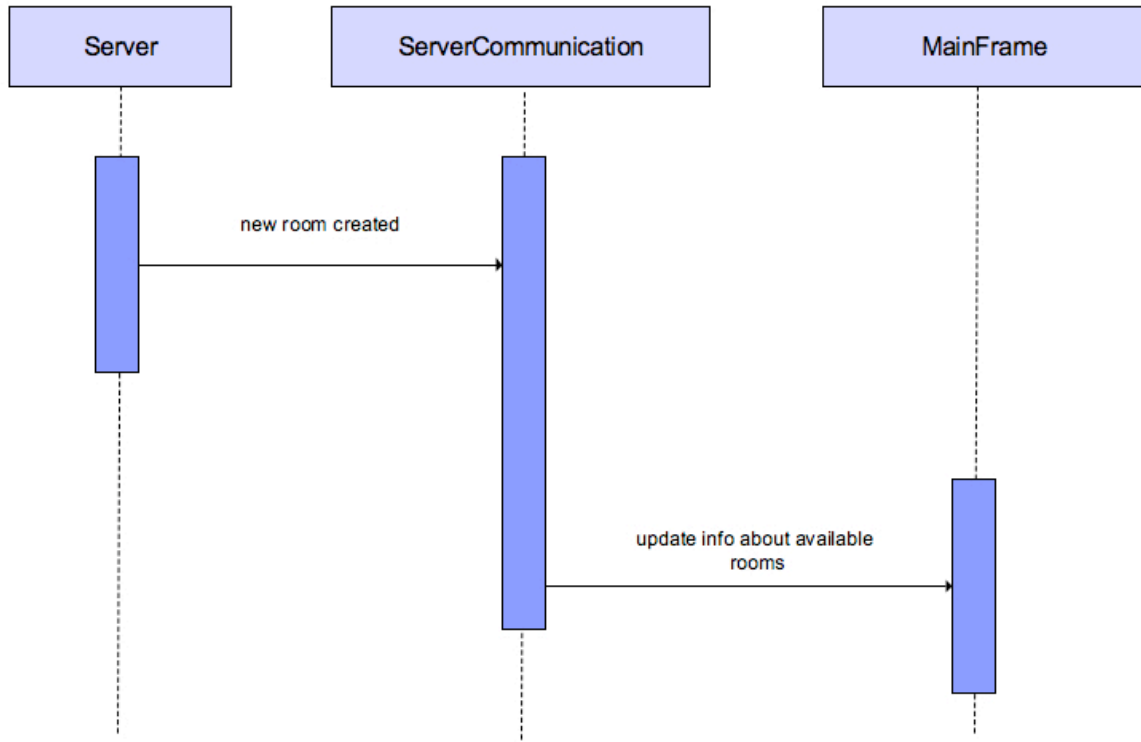
Create room sequence diagram



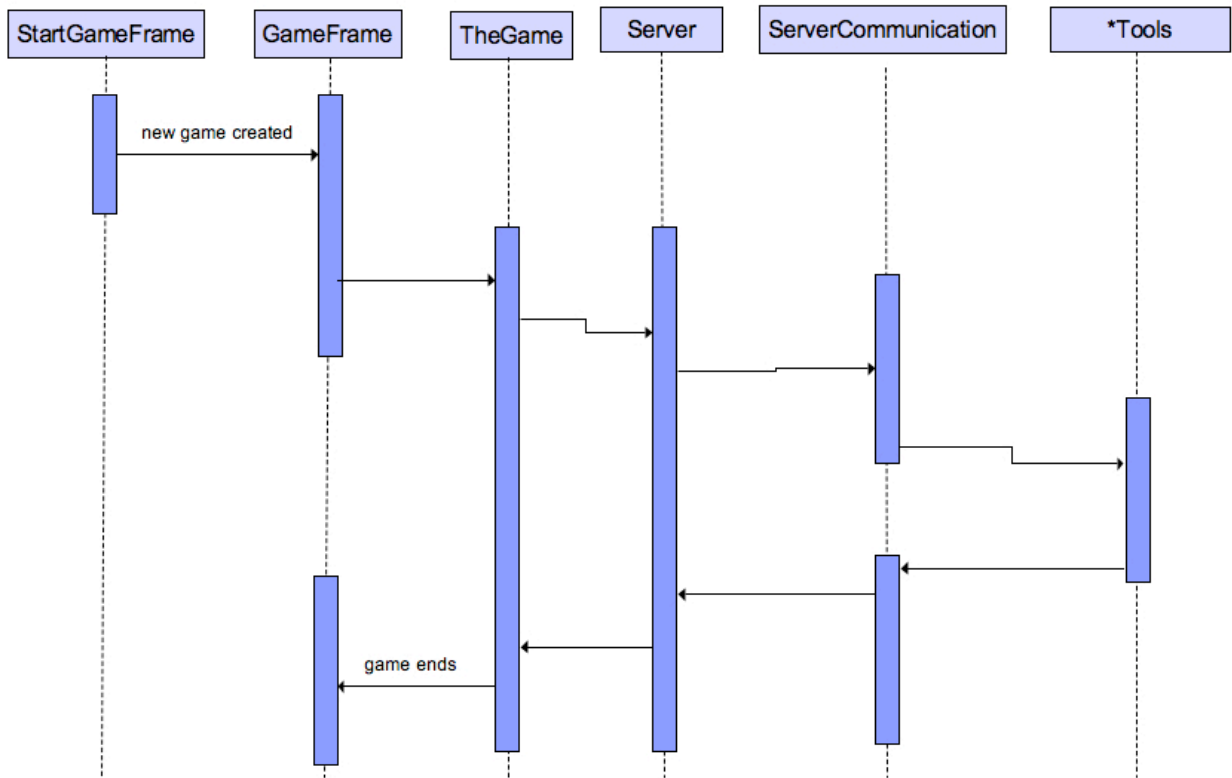
Join room sequence diagram



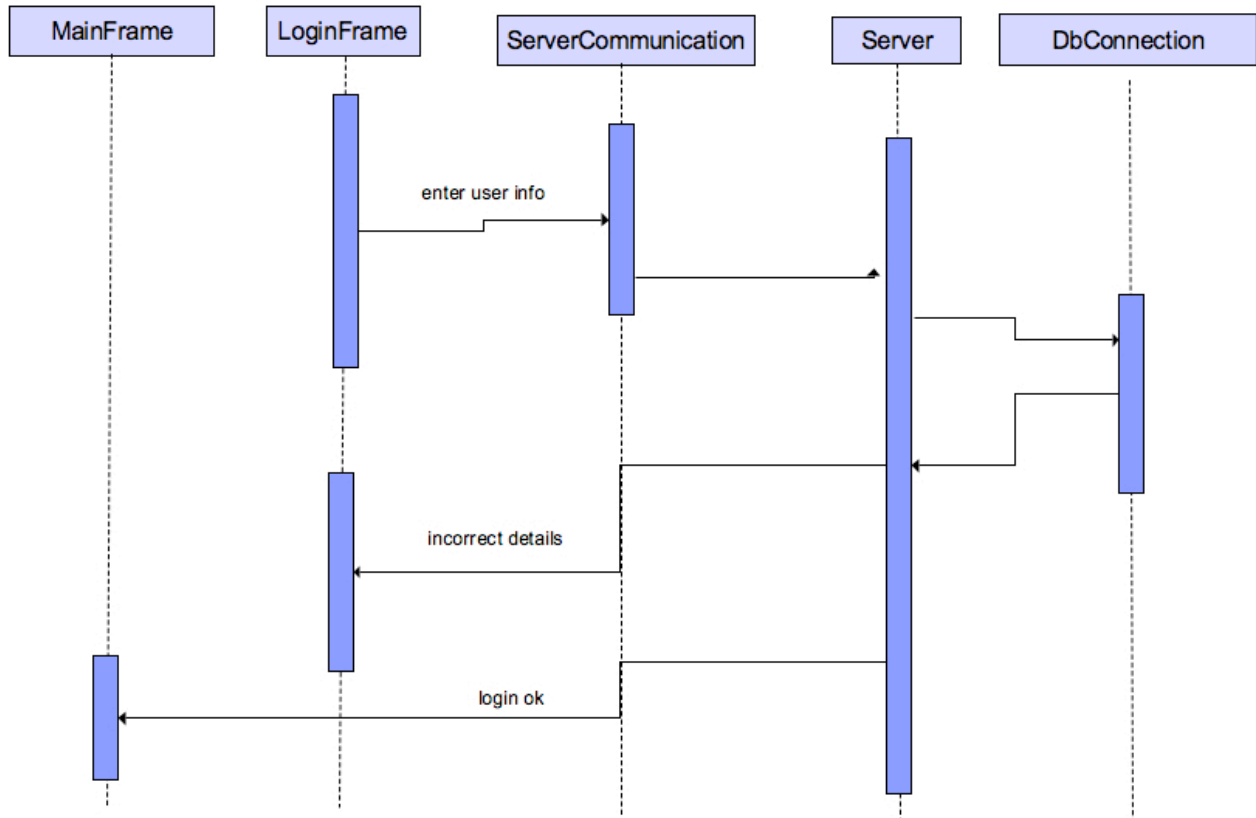
Available room sequence diagram



The game sequence diagram



Login sequence diagram



5.5 Detailed Design

Database design:

We are going to use a small database. It is going to look something like this. The first one is just for the login scenario. The rest of the information about the user is stored on the server.

The other database stores the words used in the game.

User(string) primary key	password(string)

--	--

word(string) primary key	category(string)

Methods:

SendChat	
Description	Sends strings that are to be shown to the other users in the chat
Parameters	strings
Return value	
Pre-conditions	A server connection and a ongoing game.
Post-conditions	

Chat	
Description	A chat field that receives strings from users.
Parameters	strings
Return value	
Pre-conditions	A server connection and a ongoing game.
Post-conditions	

ChatListener	
Description	Checks for updates in the Chat and updates it
Parameters	strings
Return value	
Pre-conditions	A server connection and a ongoing game.
Post-conditions	

DrawLine	
Description	Draws a line on the canvas
Parameters	x and y parameters
Return value	x and y parameters
Pre-conditions	A server connection and a ongoing game.
Post-conditions	The line is drawn on the canvas

DrawCircle	
Description	Draws a circle on the canvas
Parameters	x and y parameters
Return value	x and y parameters
Pre-conditions	A server connection and a ongoing game.
Post-conditions	The circle is drawn on the canvas and is visible for the other players

DrawPen	
Description	Draws with a pen on the canvas
Parameters	x and y parameters
Return value	x and y parameters
Pre-conditions	A server connection and a ongoing game.
Post-conditions	The drawn parts on the canvas is visible for the other players

Guess	
Description	Sends a guess(string) that is to be checked if it is correct.
Parameters	
Return value	string
Pre-conditions	A server connection and a ongoing game.
Post-conditions	A string is send

Login	
Description	Has got 2 strings as indata. It compares the strings with the user and password in the database and returns a boolean.
Parameters	strings
Return value	boolean
Pre-conditions	Two strings, user id and password
Post-conditions	The circle is drawn on the canvas and is visible for the other players

RemoveUser	
Description	This method removes a user
Parameters	strings
Return value	
Pre-conditions	Two strings, user id and password.
Post-conditions	The user is deleted from server and database if password is correct.

AddFriend	
Description	Adds a friend to the user
Parameters	string
Return value	
Pre-conditions	
Post-conditions	A user is added to the friend list.

RemoveFriend	
Description	Removes a friend to the user
Parameters	string
Return value	
Pre-conditions	
Post-conditions	A user is removed from the friend list.

Time	
Description	Is used to check the time when a player is drawing
Parameters	
Return value	boolean
Pre-conditions	A player is drawing
Post-conditions	After a specific period of time a boolean has been send.

CanvasBroadcast	
Description	Sends the updates from the canvas to the clients
Parameters	coordinates
Return value	
Pre-conditions	The canvas is updated
Post-conditions	The updates on the canvas are visible to the users

ClearCanvas	
Description	When called is clears the canvas
Parameters	
Return value	
Pre-conditions	
Post-conditions	The canvas is wiped clear

Check	
-------	--

Description	Compares two strings to see if a guess is correct
Parameters	string
Return value	boolean
Pre-conditions	
Post-conditions	Returns true if a guess is correct and false if not.

CreateRoom	
Description	Creates a new game room. Opens a GameFrame.
Parameters	
Return value	
Pre-conditions	User is logged in and in the StartFrame
Post-conditions	A new GameFrame is opened

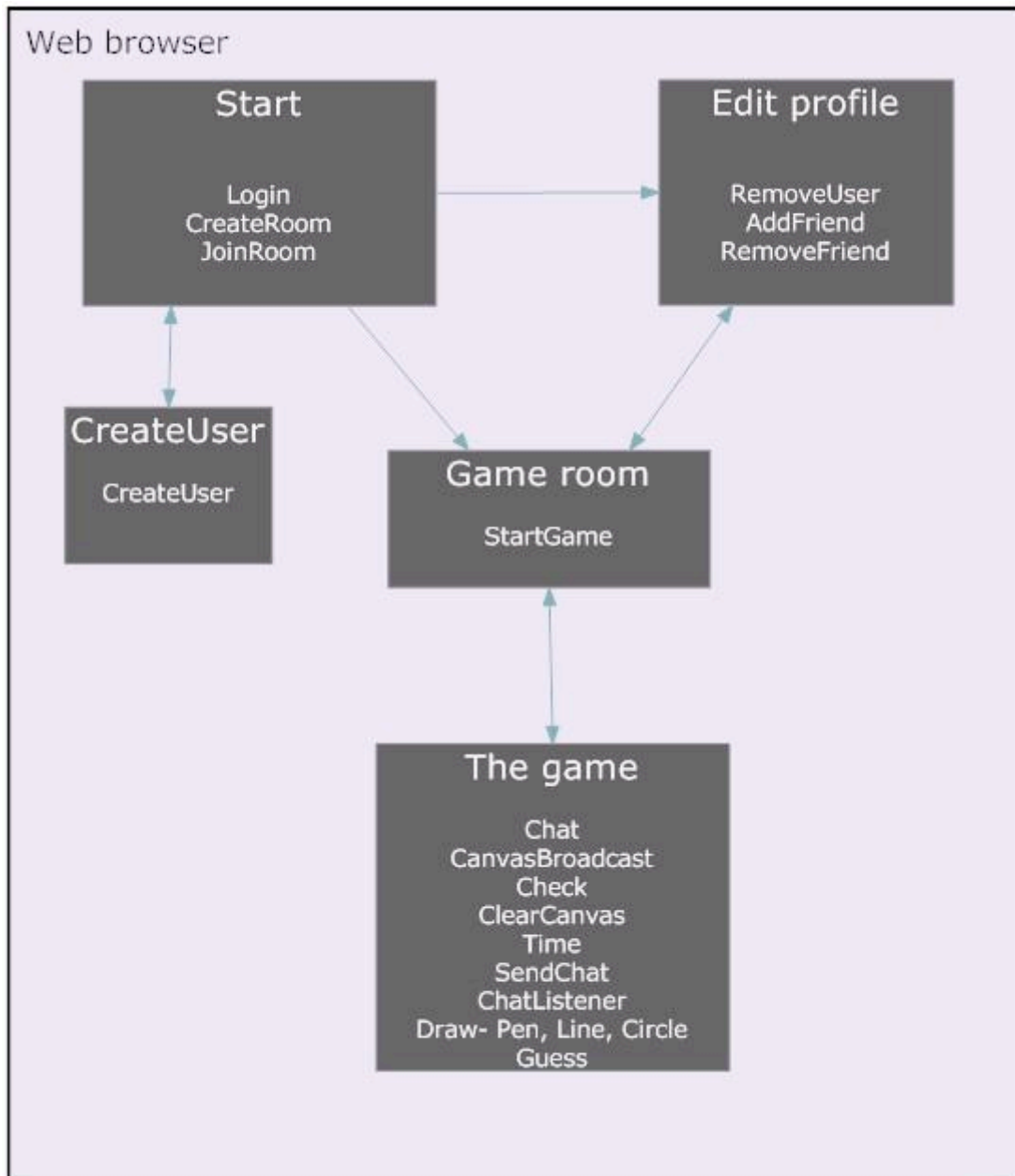
JoinRoom	
Description	This method is used when a user joins a game room
Parameters	
Return value	
Pre-conditions	A logged in user tries to join a room
Post-conditions	If the room is not full the user joins it

StartGame	
Description	Starts a game with all the users in the game room
Parameters	
Return value	
Pre-conditions	Enough players are in the game room
Post-conditions	A game is begins

CreateUser	
------------	--

Description	This method creates a user
Parameters	strings
Return value	
Pre-conditions	User information has been entered
Post-conditions	A new user is added in the user database and information about him/her is stored on the server.

5.6 Package Diagram



6. Functional Test Cases

6.1 General Game

6.1.1

Function being tested:

A player shall be able to create a game.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user has logged in to the system and is currently located in the “start frame”.

Input:

The user clicks on “Create Room”.

Expected Output:

A new window appears where you play the actual game.

Instructions:

1. Click on “Create Room”.
2. Set game preferences.
3. Click on “Start Game”.

6.1.2

Function being tested:

A player shall be able to join a game.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user has logged in to the system and is currently located in the “start frame”.

Input:

The user clicks on “Join Room”.

Expected Output:

A new window appears where you can join one of two teams and wait for the room creator to start the game.

Instructions:

1. Click on “Join Room”.
2. Join a team.
3. Wait for game creator to start the game.

6.1.3

Function being tested:

A Player shall be able to leave a game.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user has logged in to the system and is currently located in a game room.

Input:

The user clicks on “Leave Room”.

Expected Output:

The current window is closed and the user is returned to the “Start Frame”.

Instructions:

1. Click on “Leave Room”.

6.1.4

Function being tested:

The system shall store profile information.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and located at the profile page.

Input:

The user clicks on “Edit profile”.

Expected Output:

The user profile shall now be saved.

Instructions:

1. Edit user profile.
2. Click on “Save Settings”.
3. Verify by entering your profile page that the profile is showing.

6.1.5

Function being tested:

You can click on a persons name to see his/her profile provided that the person is a member.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and located in the “Start Frame”.

Input:

The user clicks on a persons name in the user list.

Expected Output:

A new window appears showing the profile of the clicked person.

Instructions:

1. Click on a persons name in the user list.

6.1.6

Function being tested:

Registered members shall be able to edit a profile page containing personal information about themselves.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

A registered member is logged in and located in the “Edit Profile Page”.

Input:

The user edits user information.

Expected Output:

The new changes to the profile are now shown in the profile page.

Instructions:

1. Edit some user information.
2. Click on “Save Settings”.

6.1.7

Function being tested:

You have to be at least four players to play; there are only two teams in one game session.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and located in a game room.

Input:

-

Expected Output:

As long as there are fewer than four players present in the game room you are unable to start the game. When in the game room you have only two teams to choose from.

Instructions:

1. Verify that the “Start Game”-button is shaded while there are fewer than four players present in the game room.

2. Verify that there are only two teams to choose from.

6.1.8

Function being tested:

There has to be at least two players in each team.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and located in a game room.

Input:

-

Expected Output:

As long as there are fewer than two players in each team you are unable to start the game.

Instructions:

1. Verify that the “Start Game”-button is shaded while there are fewer than two players in each team.

6.1.9

Function being tested:

The system evens the teams out if a player exits a game making the teams uneven.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and playing a game.

Input:

A player leaves the game making the teams uneven.

Expected Output:

When a player leaves a team making one team have more than one player more on their team than the other team, one player from that team joins the team with fewer players.

Instructions:

1. Play a game with one team with n players and the other team with $n+1$ players.
2. See to it that one player from the team with n players leaves the game.
3. Verify that the system then moves one player from the $n+1$ team over to the other team.

6.1.10

Function being tested:

If a player is inactive for a specific period of time he is kicked out of the game.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and playing a game.

Input:

-

Expected Output:

The player who is being inactive is kicked from the game.

Instructions:

1. Be inactive in a game
2. Verify that you are kicked from the game.

6.1.11

Function being tested:

A dice will tell the teams how many steps on the board they should move.

Requirement document reference:

Functional requirements, page 9

Initial system state:

The user is logged in and playing a game.

Input:

The user guesses the correct word.

Expected output:

The guess word is correct; you will receive a “Your guess is correct!” output on the display.

The team’s piece on the game board will move as many steps shown by the dice.

Instructions:

1. Choose the “Game chat window” by clicking in it
2. Type in a word and press “Enter”
3. Verify that “Your guess is correct!” output is showing.
4. Verify that the piece on the game board moves as many steps as shown by the dice.

6.1.12

Function being tested:

There shall be a game board that contains the path from finish to goal, where the first team that gets to the goal wins.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and is playing a game.

Input:

-

Expected Output:

In the game window there is a game board displayed.

Instructions:

1. Start a game.
2. Verify that a game board is showing.

6.1.13

Function being tested:

The squares in the path shall have a color corresponding to a category of words.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and playing a game.

Input:

-

Expected Output:

The game board's squares have different colors.

Instructions:

1. Start a game.
2. Verify that the squares of the game board have different colors.

6.1.14

Function being tested:

There shall be two kinds of guessing squares a team can land on. Double guess where both teams have a chance to guess the right word and single guess where only the team that lands on the square may guess.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and playing a game.

Input:

-

Expected Output:

The game board contains two kinds of squares, one for single guess and one for double guess.

Instructions:

1. Start a game.
2. Verify that the game board contains two kinds of squares.

6.1.15

Function being tested:

During the game there shall be a team chat available, where you can chat with your team members.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in and playing a game.

Input:

Keyboard input in the team chat.

Expected Output:

The game window is displaying a team chat where messages from other teammates are displayed.

Instructions:

1. Write something in the team chat window.
2. Verify with another user in the game not on your team that what you wrote is not visible to him.
3. Verify with a user on your team that what you wrote is visible to him.

6.1.16

Function being tested:

A tutorial with complete instructions for the game is available before and during game play.

Requirement Document reference:

Functional Requirements, page 9.

Initial system state:

The user is logged in to the system.

Input:

The user clicks on the tutorial button.

Expected Output:

A new window appears showing the game rules and a tutorial.

Instructions:

1. Click on the tutorial button.
2. Verify that a window with game rules and a tutorial is displayed.

6.1.17

Function being tested:

It is possible to use a microphone.

Requirement Document reference:

Functional Requirements, page 10.

Initial system state:

The user is logged in to the system.

Input:

Sound from a microphone.

Expected Output:

Sounds from the microphone of other players are played out of the speakers.

Instructions:

1. Talk in your microphone
2. Listen to other players talking in their microphones.

6.1.18

Function being tested:

The ability to store the drawn picture for later use.

Requirement Document reference:

Functional Requirements, page 10.

Initial system state:

The user is logged in and playing a game.

Input:

A drawn picture.

Expected Output:

The picture is saved on your computer.

Instructions:

1. Draw a picture in the game.
2. Save the picture.
3. Verify that the picture is stored on your computer.

6.2 Drawing

6.2.1

Function being tested:

The team members take turns drawing. You cannot choose which one who should draw.

Requirement document reference:

Functional requirements p.10

Requirements to test the function:

Play enough number of guesses, to see that the team member's takes turn in drawing.

Initial system state:

The game has started

Input:

Keyboard input from guessing of the word in the Game chat window

Mouse input when drawing a word.

Expected output:

The client display will show the user typed letters on the screen.

If the guess word is correct, you will receive a “Your guess is correct!” output on the display.
Another team member will be up next to draw, the order cannot be changed

Instructions:

1. Choose the “Game chat window” by clicking in it
2. Type in a word and press “Enter”
3. Verify that “Your guess is correct!” output is showing.
4. Verify that another team member’s is drawing the next word.
(The order cannot be changed)
5. If it is the users turn to draw, draw the word
6. Verify that a different drawer will be up next
7. Repeat the above steps over again
8. Verify that the same order of players take turns in drawing

6.2.2

Function being tested:

The player will draw inside of a canvas that everybody can see.

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started and it is your time to draw the word.

Input:

Mouse input when drawing the word.

Expected output:

Display everything that the drawer has drawn.

Every client will display the drawn word.

Instructions:

1. Choose the “Drawing window” by clicking in it
2. Draw something by choosing a specific tool in the toolbox
3. Verify that the drawn figure is shown.
(Both on the drawers display and on the other players display)

6.2.3

Function being tested:

Before the drawer starts drawing the system shall play a specific sound to let everybody know that the drawing begins.

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started and it is time for someone to draw a word.

Input:

-

Expected output:

A specific sound from the speakers

Instructions:

1. Listen to the specific sound

6.2.4

Function being tested:

When a player is drawing, chat and use of the microphone for that player is disabled.

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started and it is time for the user to draw a word.

Input:

Mouse to draw the word

Microphone to talk to other players

Expected output:

Display everything that the drawer has drawn.

No sound from the drawer will be sounded on another client's computer.

Typing in the Game chat window and Team chat window is disabled.

Instructions:

1. Try to talk into the drawers computer microphone
2. Verify on another client's computer that the drawer's microphone is muted.
3. Try to type in the Game chat window or in the Team chat window on the drawer's client.
4. Verify that both chat windows are disabled.

6.2.5

Function being tested:

While drawing it is possible to choose colors and choose between different set of drawing functions, such as: circle, straight line, rectangle/box, eraser and a pen with different thicknesses.

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started and it is time for the user to draw a word.

Input:

Using the mouse to draw the word and choose between the different drawing tools.

Expected output:

Display everything that the drawer has drawn.

Instructions:

1. Click in the Drawing area.
2. Choose between the different drawing tools.
3. Draw using the chosen drawing tool.
4. Chang drawing tool.
5. Verify that it works to change between the drawing tools.

6. Repeat instruction 2-6 to verify that every drawing tools works.

6.3 Guessing

6.3.1

Function being tested:

One can only guess the correct word by typing it in the Game chat window.

Requirement document reference:

Functional requirements p.10

Requirements to test the function:

Guessing of the word is supposed to be done in the Game chat window.

The user must know the correct word.

Initial system state:

The game has started.

Input:

Keyboard input from guessing of the word in the Game chat window.

Expected output:

The client display will show the user typed letters on the screen.

If the guess word is correct, you will receive a “Your guess is correct!” output on the display.

Instructions:

1. Choose the “Game chat window” by clicking in it
2. Type in a word and press “Enter”
3. Verify that “Your guess is correct!” output is showing.

6.3.2

Function being tested:

You have one minute to successfully guess the correct word that your team member is drawing. If you are unable to guess correctly it is the opposing teams turn to draw.

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started. It is your teams turn to guess the word that one of your team members is drawing

Requirements to test the function:

Guessing of the word is supposed to be in the Game chat window. Not guessing the correct word within one minute.

Input:

Keyboard input from guessing of the word in the Game chat window

Expected output:

The client display will show the user typed letters on the screen.

The guessed word is incorrect; nothing will happen.

After a minute has passed, you will receive a message that the time is out and that it is the opposing teams turn to draw and guess a new word.

The time is counting down from 60 seconds and is displayed close to the drawing.

Instructions:

1. Choose the “Game chat window” by clicking in it
2. Type in a word and press “Enter”
3. Verify that nothing happened expect that the earlier typed word and other players chat messages are shown in the Game chat window.
4. Verify that the countdown timer is counting down correctly.
5. Verify that it is the opposing teams turn to draw and guess the word.

6.3.3

Function being tested:

Every guess is matched against the correct word. For the answer to be correct it has to be completely correct. Use of upper or lower case is ignored.

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started

It is your team’s turn to guess the word that one of your team members is drawing

Requirements to test the function:

Guessing of the word is supposed to be in the Game chat window. The user must know the correct word and type in both upper and lower case letter of the word.

Input:

Keyboard input from guessing of the word in the Game chat window

Expected output:

The client display will show the user typed letters on the screen.

The guess word is correct; you will receive a “Your guess is correct!” output on the display.

Instructions:

1. Choose the “Game chat window” by clicking in it
2. Type in a word and press “Enter”
3. Verify that “Your guess is correct!” output is showing.

6.3.4

Function being tested:

If your answer is correct, you may proceed in the game by rolling a dice and move as many steps as shown by the dice (on the game board).

Requirement document reference:

Functional requirements p.10

Initial system state:

The game has started,

It is your team’s turn to guess the word that one of your team members is drawing

Requirements to test the function:

Guessing of the word is supposed to be in the Game chat window. The user must know the correct

word.

Input:

Keyboard input from guessing of the word in the Game chat window

Expected output:

The client display will show the user typed letters on the screen.

The guess word is correct; you will receive a “Your guess is correct!” output on the display.

Instructions:

The team’s piece on the game board will move as many steps shown by the dice.

1. Choose the “Game chat window” by clicking in it
2. Type in a word and press “Enter”
3. Verify that “Your guess is correct!” output is showing.
4. Verify that the piece on the game board moves as many steps as shown by the dice.