Project Requirements

Project Name: Wordfinder

Team: 4

Customer: Andrei Chis

Revision History

|  |  |  |
| --- | --- | --- |
| Version | Date | Revision Description |
| 1.0 | 02.10.2013 | Rough sketch |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Date: October 02, 2013

Index

1 Introduction 4

1.1 Purpose 4

1.2 Stakeholders 4

1.3 Definitions 4

1.4 Overview of Document 4

1.5 Overview of Document 5

1.6 References 5

2 Overall description 6

2.1 Diagram 6

2.2 Use cases 7

2.2.1 1. Play Game 7

2.2.2 2. View Stats 7

2.2.3 3. Retry Board 8

2.2.4 4. Set up social function 9

2.2.5 5. View Friend List 10

2.2.6 6. Add Friend 10

2.2.7 7. Remove Friend 11

2.2.8 8. View Dictionary 12

2.2.9 9. Edit Dictionary 13

2.2.10 10. Add Dictionary 14

2.2.11 11. Remove Dictionary 14

2.2.12 12. Play Friend's board 15

2.2.13 13. Accept friendship 16

2.2.14 14. Edit Settings 16

2.3 Actor characteristics 17

3 Specific Requirements 18

3.1 Functional Requirements 18

3.1.1 Use case 1. Play Game 18

3.1.2 Use case 2. View stats 18

3.1.3 Use case 3. Retry board 18

3.1.4 Use case 4. Set up social function 18

3.1.5 Use case 5. View friends List 18

3.1.6 Use case 6. Add friend 18

3.1.7 Use case 7. Remove friend 19

3.1.8 Use case 8. View dictionary 19

3.1.9 Use case 9. Edit dictionary 19

3.1.10 Use case 10. Add dictionary 19

3.1.11 Use case 11. Remove Dictionary 19

3.1.12 Use case 12. Play friends board 19

3.1.13 Use case 13. Accept Friendship 19

3.1.14 Use case 14. Edit Settings 19

3.2 Nonfunctional Requirements 20

3.2.1 Use case 1. Play Game 20

3.2.2 Use case 2. Show stats 20

3.2.3 Use case 3. Retry board 20

3.2.4 Use case 4. Set up social function 20

3.2.5 Use case 5. Show friends list 20

3.2.6 Use case 6. Add friend 20

3.2.7 Use case 7. Remove friend 21

3.2.8 Use case 8. View dictionary 21

3.2.9 Use case 9. Edit dictionary 21

3.2.10 Use case 10. Add dictionary 21

3.2.11 Use case 11. Remove dictionary 21

3.2.12 Use case 12. Play friends board 21

3.2.13 Use case 13. Edit settings 21

3.2.14 Use case 14. Accept Friend request 21

3.3 General 21

# Introduction

## Purpose

The reason this document is written, is for the stakeholders and the developers to define an appropriate definition of the Android application “Wordfinder”. It should declare the application’s purpose and features, including the use cases and requirement specifications.

## Stakeholders

* User
* Client
* Developer

## Definitions

|  |  |
| --- | --- |
| Board / Matrix | Area where the player has to play in, the place where he has to find the words. |
| Dictionary |  |
| Friend | Another player who was added as “Friend” in the social section of the game. A friend allows to play other modi of the game |
| Highscore | Amount of all points, that a player collected in his playtime. |
| Language | Defines the language in which the game is played (language of the words). |
| Player | The person who uses the application. |
| Time, -r | A visual countdown that is used to determine the length of the game. |
| Word | Predefined chain of letters |
| Wordfinder | Name of the Android-application |
|  |  |

**“Synchronization, if possible”:** The reoccurring formulation “synchronization, if possible” describes the following: When the app synchronize, it tries to upload the player’s scores to the server[[1]](#footnote-1) and to download the friend’s scores from the server. This data is stored in the **local storage**. If the upload/download fails for whatever reason (e.g. no / bad internet connection, server not responding) this step is simply skipped.

**“Local Storage”:** The local storage refers to the locally stored recordings of the player’s and the friend’s scores and boards. All interaction with scores are done over this storage. That way the game’s functionality can to large parts be guaranteed even if there’s no internet connection.

## Overview of Document

The next chapter, the Overall description, contains a diagram that present an overview and summary of all the use cases. Therefor the Overall description describes the application in the sight of an user, and simulates every action he should be able to perform.

The focus of the third chapter is primary written for the developers and describes in technical terms and the details of the functionalities of the application.

## Overview of Document

The next chapter, the Overall description, contains a diagram that present an overview and summary of all the use cases. Therefor the Overall description describes the application in the sight of an user, and simulates every action he should be able to perform.

The focus of the third chapter is primary written for the developers and describes in technical terms and the details of the functionalities of the application.

## References

None atm.

# Overall description

## Diagram

## Use cases

### 1. Play Game

Actors

* Player (Bob)

Description

* Bob wants to play a local game

Trigger

* Presses the 'Start Game' button in the main screen

Pre-Conditions

Post-Conditions

1. New entry in personal recordings (Score, Board, ...)
2. Push personal recordings to server, if possible

Main scenario

1. Bob starts the 'word finder' app
2. Bob presses on 'start game' in the main screen
3. The System generates a new game
   1. The system tries to use as many words form the selected dictionary as possible[[2]](#footnote-2)
   2. The remaining board is filled up with words from the default dictionary (English)
4. Bob plays and finished the game
   1. The game is finished once Bob has found all the words in the board, or the timer[[3]](#footnote-3) has expired.
5. Score is displayed and saved in the personal recordings
6. The system tries to synchronize the personal recordings with the server, if possible
7. returning to the main screen

Alternative scenarios

1. Game is interrupted (e.g. incoming call) or paused manually
2. Pause the game
3. Hide the board, show 'resume' button
4. Wait until 'resume' is pressed
5. resume the game and continue normal execution
6. Game is quit manually

* End game and return to main screen

Special Requirements

* For synchronization with the server, a standing internet connection is required.

Notes

* The game can only be used to it's full extend when an internet connection is present, but local games can be played without an internet connection.

### 2. View Stats

Actors

* Player (Bob)

Description

* Bob want's to view his or a friend's score.

Trigger

* Bob presses ‘Scores’, if he wants to see his own scores
* Bob selects a friend from his friend list, if he wants to see his friends scores

Pre-Conditions

Post-Conditions

Main scenario

1. Bob selects 'Scores' for his own scores or selects a friend from the friend list if he wants to see that friends scores
2. The app tries to synchronize with the server, if possible
3. Bob is presented with a list with the top 5 and most recent 20 boards of his or the friend he selected
4. Under each board the personal score and the top score of all friends is shown. When viewing a Friend Board list, the friends score is also displayed.
5. Bob selects a board from the list.
6. The boards detailed stats are shown, including the board setup[[4]](#footnote-4), the timer length and all the recorded boards from Bob’s friends.

Alternative scenarios

Special Requirements

* For synchronization with the server, a standing internet connection is required.

Notes

### 3. Retry Board

Actors

* Player (Bob)

Description

* Bob wants to play a local game he already played before, and that is visible in his or a friend’s scores.

Trigger

* Bob selects 'Retry' on one of his scored boards.

Pre-Conditions

* Bob is viewing his or a friend’s scores and has selected a board.

Post-Conditions

1. New entry in personal recordings (Score, Board, ...)
2. Push personal recordings to server, if possible

Main scenario

1. Bob selects 'retry'
2. The board is loaded
3. The board's dictionary is loaded
4. The game is generated. The loaded board is used instead of a newly generated one
5. Bob plays and finished the game
6. Score is displayed and saved in the personal recordings
7. The system tries to synchronize the personal recordings with the server, if possible
8. returning to the main screen

Alternative scenarios

1. Game is interrupted (e.g. incoming call) or paused manually
2. Pause the game
3. Hide the board, show 'resume' button
4. Wait until 'resume' is pressed
5. resume the game and continue normal execution
6. Game is quit manually
7. End game and return to main screen

Special Requirements

* For synchronization with the server, a standing internet connection is required.

Notes

* The game can only be used to it's full extend when an internet connection is present, but local games can be played without an internet connection.

### 4. Set up social function

Actors

* Player (Bob)

Description

* Bob wants to share his boards with his friends and play friends boards

Trigger

* Bob starts App

Pre-Conditions

* Bob hasn't set up the sharing functionality yet.

Post-Conditions

* Bobs boards are visible to others, and he can add friends to play their boards

Main scenario

1) Bob starts the 'word finder' app

2) Bob presses 'friends' in the main screen

3) Bob is asked to enter his email address and a password[[5]](#footnote-5) for creating an account.

4) Bob now has the social feature set up.

Alternative scenarios

1. The Device is not connected to the internet
   * When clicking on friends, Bob gets a message which states that Bob has to be connected to the internet to interact with his friends

Special Requirements

* Phone has to be connected to the internet

Notes

* When the social function is set up, anybody with Bobs email address can follow his plays

### 5. View Friend List

Actors

* Player (Bob)

Description

* Bob wants to see an overview of all his friends

Trigger

* Bob selects ‘friends’ in the main screen

Pre-Conditions

* Bob has the sharing function set up.

Post-Conditions

Main scenario

1. Bob presses on 'friends' in the main screen
2. The app synchronizes with the server, if possible
3. Bob sees the list with all his friends

Alternative scenarios

1. Bob hasn’t setup the social function yet
   1. Bob is asked to set the social function up according to usecase 4.

Special Requirements

Notes

### 6. Add Friend

Actors

* Player (Bob), Friend (Susie)

Description

* Bob wants to add Susie to his friend list.

Trigger

* Bob starts App

Pre-Conditions

* Bob has the sharing function set up.
* Bob’s Device is connected to the Internet.

Post-Conditions

* New friend added to the friends list

Main scenario

1. Bob starts the 'word finder' app
2. Bob presses on 'friends' in the main screen
3. Bob sees the list with all his friends
4. Bob presses ‘add friend’
5. Bob now enters Susie's email address.
6. Bob now sent Susie a ‘friend request’
7. As soon as Susie accepts the ‘friendship’, Bob can follow her – and she can follow his – game history.

Alternative scenarios

1. The Device is not connected to the internet
   * When Pressing on ‘Add Friend’ a message is shown to inform Bob, that he has to be connected to the Internet to add a ‘friend’
2. Susie does not play Wordfinder
   * After entering the email address, a ‘player not found’ message is shown (?)[[6]](#footnote-6)
   * Bob can enter a new email address and try again, or he can abort the adding process and return to the friends overview

Special Requirements

* Bob’s Phone has to be connected to the internet

Notes

* The friendship is only established once Susie accepts the friendship until then no changes to Bob’s friends view are made

### 7. Remove Friend

Actors

* Player (Bob), Friend (Susie)

Description

* Bob wants to remove Susie from his friend list.

Trigger

* Bob starts App

Pre-Conditions

* Bob has the sharing function set up.
* Bob has Susie in his friend-list.
* Bob is in the friends list
* Bob needs to be connected to the internet

Post-Conditions

* Susie is removed from the friends list with all her boards and scores.

Main scenario

1. Bob deletes Susie by directly deleting her from the list
2. Bob is asked to confirm the deletion of Susie as friend
3. Bob confirms the deletion
4. The app synchronizes with the server, wich removes all of Susie’s data from Bob’s phone.

Alternative scenarios

1. The Device is not connected to the internet
   1. Bob is informed, that he has to be connected to the internet for this action.

Special Requirements

* Phone has to be connected to the internet

Notes

* Bob’s data is also deleted from Susie’s phone with her next synchronization

### 8. View Dictionary

Actors

* Player (Bob)

Description

* Bob wants to view one of his dictionaries

Trigger

* Bob starts App

Pre-Conditions

Post-Conditions

Main scenario

1. Bob starts the 'word finder' app
2. Bob presses ‘settings’ 🡪 ‘dictionaries’
3. A list with all his dictionaries is shown.
4. Bob selects the one he wants to see
5. Bob is presented with a list of all word in his dictionary
6. Bob chooses an existing language (German in our case)
7. Bob also has a search option to find a specific word in the dictionary

Alternative scenarios

Special Requirements

Notes

### 9. Edit Dictionary

Actors

* Player (Bob)

Description

* Bob wants to edit a dictionary of his.

Trigger

* Bob selects ‘edit’ in the dictionary content view (word list)

Pre-Conditions

- Bob has to be in the dictionary content view as described in usecase 7.

Post-Conditions

* The content of the dictionary is updated.

8.6 Main scenario

1. Bob selects ‘edit’ in the dictionary content view.
2. Bob enters the editing view. In this view he sees a list with all the word.
3. Bob edits the dictionary:
   1. The words can be deleted by removing them from the list.
   2. New words can be added by typing them into the text field on top of the list. This function searches to base dictionary (English) for the entered word continuously. When words are found that match the pattern[[7]](#footnote-7), they are displayed in a popup list below the text are and can be added to the List by pressing on them.
4. Bob saves the dictionary by pressing on ‘save’. This action brings him back to the dictionary content view.

Alternative scenarios

1. Bob doesn’t want to save his changes to the dictionary
   1. Bob presses on ‘back’. This brings him back to the dictionary content view without saving his changes.

Special Requirements

Notes

### 10. Add Dictionary

Actors

* Player (Bob)

Description

* Bob wants to add a new dictionary and add words to it.

Trigger

* Bob starts App

Pre-Conditions

-

Post-Conditions

* A new dictionary with some words is created and can be used for a game.

8.6 Main scenario

1) Bob starts the 'word finder' app

2) Bob presses ‘settings’ 🡪 ‘dictionaries

3) Bob selects the ‘add new’ option and is presented with the request to enter a name

4) Bob enters a name for the dictionary and selects ‘continue’

5) Bob is presented with the dictionary editing view as defined in usecase 8.

Alternative scenarios

1. Bob doesn’t enter a valid name[[8]](#footnote-8)

Special Requirements

-

Notes

-

### 11. Remove Dictionary

Actors

- Player (Bob)

Description

- Bob wants to remove a dictionary

Trigger

- Bob starts App

Pre-Conditions

-

Post-Conditions

- A dictionary has been removed

Main scenario

1. Bob starts the 'word finder' app
2. Bob presses language.
3. Bob presses the 'edit' button
4. Bob selects the dictionary to delete and confirms
5. The dictionary is deleted and Bob gets back to the languages menu

Alternative scenarios

Special Requirements

Notes

### 12. Play Friend's board

Actors

- Player (Bob)

Description

- Bob wants to play a game a friend has played before, and that is visible the friend's scores.

Trigger

- Bob selects 'Play' on one of his friend's boards.

Pre-Conditions

- Bob is viewing his friend's board details

Post-Conditions

1. New entry in personal recordings (Score, Board, ...)
2. Push personal recordings to server, if possible

Main scenario

1. Bob selects ‘Play’
2. The board and it’s timer length are loaded
3. The game is generated. The loaded board is used instead of a newly generated one
4. Bob plays and finished the game
5. Score is displayed and saved in the personal recordings
6. system tries to synchronize the personal recordings with the server, if possible
7. to the main screen

Alternative scenarios

1. Game is interrupted (e.g. incoming call) or paused manually
   1. Pause the game
   2. Hide the board, show ‘resume’ button
   3. Wait until ‘resume’ is pressed
   4. resume the game and continue normal execution
2. Game is quit manually
   1. End game and return to main screen

### 13. Accept friendship

Actors

- Player (Susie), Friend (Bob)

Description

- Bob has sent Susie a Friend request and she accepts.

Trigger

- On Synchronization data about the friend request is received.

Pre-Conditions

Bob has sent Susie a friend request

Post-Conditions

Susie is now friends with Bob

Main scenario

* + - 1. Susie’s App successfully synchronizes and receives data about Bob’s friend request
      2. Susie is informed that Bob has sent her a friend request. For identification, Bob’s email is shown.
      3. Susie now has the option to accept or decline that friend request
      4. Susie accepts the friendship
      5. Bob is added to Susie’s friend list, and Susie is added to Bob’s.
      6. The app synchronizes – if possible – to get Bob’s data
      7. Bob is informed about the acceptance of the friendship[[9]](#footnote-9)

Alternative scenarios

Susie declines the request.

* + 1. The request is deleted and Bob is informed about the fact that his request has been declined.[[10]](#footnote-10)
    2. There will be no changes to Susie’s or Bob’s data

### 14. Edit Settings

Actors

- Player (Bob)

Description

- Bob wants to change his settings

Trigger

- Bob presses on ‘Settings’ in the main screen

Pre-Conditions

Post-Conditions

Main scenario

1. Bob presses on settings
2. Bob now has the option to change his game timer settings and his dictionaries.
   1. The game timer setting can be changed directly in the settings screen
   2. The Dictionary settings are entered by pressing ‘Dictionaries’
3. Bob presses on ‘Dictionaries’
4. Bob is now presented with a List of his dictionaries.
5. Bob selects the dictionary he wants to play with.
6. Bob now is in the same view as when viewing a dictionary
7. Bob can select ‘Use Dictionary’.
8. From now on this Dictionary is the main one.

Alternative scenarios

Notes

## Actor characteristics

Main actor of our use cases in Bob.

Bob is a informatics student at the university of Bern.

So Bob is experienced using software, and acquiring new skills.

Bob has owned Android phones since the first Android came out.

Bob so has a lot of experience in using Android software.

Second actor is Susie.

Susie has already installed and used our app.

# Specific Requirements

## Functional Requirements

### Use case 1. Play Game

* Makes playing Word Finder game.
* Home-screen has a start game button to start game.
* User makes his profile at first opening, user database is created.
* Play game initializes a new 7x7 word game grid, sets time and score to initial value. (minimal word-count of 10 words and minimal word-length of 3 letters are assured).
* Initial value of score is 0.
* Initial value of time is the time set to the value defined in the settings.
* Internet connection for synchronization.
* Pause button stops the game (time halts) and shows pause screen.
* Pause screen shows a resume button.
* At end of game time and score is showed to player, and achievement of new hight score is shown to player.

### Use case 2. View stats

* Shows own scores.
* Shows friend’s scores.
* Score button in main screen.
* Friend’s Scores accessed over friends menu

### Use case 3. Retry board

* Play a known board once more.
* Retry button beside the scores of played board.
* A retrial count as a normal game, and thus manipulates the board recordings in the same way.

### Use case 4. Set up social function

* Setting up sharing functionality means uploading profile to server.
* Server is needed for distribution and maintenance of user data.
* Friend button available in home screen.
* Login window : 1 Username , 2 Password.
* App can establish a internet connection to a determined server.
* Capability of sending and receiving data to/from internet.
* Capability of local storage of the last known friends’ data.

### Use case 5. View friends List

* Show a list of the user’s friends
* Used to access friend’s details

### Use case 6. Add friend

* App allows to add friends to friend list.
* In the Friends overview friends can be added over the ‘+’ button.
* Pressing on the ‘add friend’ button sends friend request to other user.
* User not added to list until acceptance
* Identify user according to selected e-mail addresses.
* Existence of a list of friends and a list of all users on server side.

### Use case 7. Remove friend

* Friend removed from table showing the friends in the friends overview.
* Deleted friend doesn`t becomes a notification when being deleted.
* Deleted friend appears in normal person list of the server.
* No data of removed friends is kept on users phone

### Use case 8. View dictionary

* App includes the standard English dictionary.
* Possibility to select own word of provided dictionary for use in own dictionary.
* Possibility to delete own word out of own dictionary.
* Create a new dictionary with own name.

### Use case 9. Edit dictionary

* Allows the user to add or remove words from dictionary. Only words from the provided (english) dictionary can be added

### Use case 10. Add dictionary

* Add dictionary with own name.
* Directly bringst he user to use case 9

### Use case 11. Remove Dictionary

* Removes dictionary from the game and the phone.

### Use case 12. Play friends board

* Play game a friend already played.
* Play button on score list of friends.

### Use case 13. Accept Friendship

### Use case 14. Edit Settings

## Nonfunctional Requirements

### Use case 1. Play Game

* <Performance requirement> : <System start (home screen) under 10s.>
* <Portability> : <System portability has to be specified before installation and download of the App.>
* <Space requirement> : <Installation space have to be specified before download and installation. (kbyte)>
* <Space requirement> : <Minimal RAM space during use has to be specified before installation. And checked for availability before usage.>
* <Space requirement> : <Space used for offline recorded scores/times/Board/ is specified and checked before playing game.>
* <Performance requirement> : <System has to stop immediately when pushing pause, the screen has to be covered in an instant.>
* <Legislative requirement> : <Copyright and terms of use has to be shown to the User before first launch of the App (during download or installation).>
* <Legislative requirement> : <User has possibility to modify or correct his own user record database.>
* <Efficiency requirement> : <Internet connection is checked when opening the game. If no connection is established a offline game will be started and game history or statistics are saved in a external file for recovery when the connection is reestablished. Assures no continuous connection searching, and performance lost by this.>
* <Performance requirement> <Opening a new game should not take more than 10s.>

### Use case 2. Show stats

### Use case 3. Retry board

* <Space requirement> : <text file for the generation numbers of the last 20 boards and the top 5.>

### Use case 4. Set up social function

* <Legislative requirement> : <User agrees terms of social networking of Word Finder Game.>
* <Security requirement> : <Using of secure script for password and user data transfers to server.>
* <Privacy requirement> : <Data is only accessible to friends and the general permission to show data to others has been accepted from the user.>
* <Product requirement> : <The phone has possibility to connect to internet: 4G, 3G,(â€¦ internet standards of mobile phones) and Wi-Fi.>
* <Space requirement> : <Space allocation for user database on server side and for offline usage on cellphone.>
* <Portability requirement> : <check if the board is playable on other device should by installing conditions.>

### Use case 5. Show friends list

### Use case 6. Add friend

* <Space requirement> : <Allocation of space for the list of friends and the list of all users on the server side ( Database ).>
* <Fault tolerance> : <A sent of a wrong friendship request cant be aborted, friendship has to be deleted once it was accepted.>
* <Privacy requirement> : <A friend can see public things.>
* <Privacy requirement> : <A other user is only a friend if friendship was accepted not yet if only a request is sent.>

### Use case 7. Remove friend

* <Legislative requirement> : <deletion of friendship don't requires double con sense.>
* <Fault tolerance> : <A sent of a wrong friendship deletion request can’t be aborted, friendship has to be recreated once it was deleted.>
* <Accessibility> : <The data of non- friends can’t be accessed.>

### Use case 8. View dictionary

* <Customizability> : <Own dictionary can be added to the game.>
* <Variability> : <Own dictionaries can variate, from user to user but standard dictionaries don't.>
* <Correctness> : <Dictionary has to contain only correct spelled words.>
* <Space requirement> : <Space for installation of dictionary databases needed.>

### Use case 9. Edit dictionary

* <redundancy> : <imported word can’t be imported if they already exists in the dictionary, duplicates are ignored and no failure is reported.>
* <privacy requirement> : <custom dictionaries are only stored locally>
* <Space requirement> : <No server-side space requirements>

### Use case 10. Add dictionary

### Use case 11. Remove dictionary

* <robustness> : <No influence on stored boards, as dictionaries are only needed for generation>
* <Failure tolerance> : <User has to confirm 2 times before deletion.>

### Use case 12. Play friends board

### Use case 13. Edit settings

### Use case 14. Accept Friend request

## General

* <Backup> : <No backup is done fore the game in execution, excepted the stats sent to the server.>
* <Backup> : <Github for version control an backups.>
* <Interoperability> : <Word Finder Game is designed to work alone, and for synchronization with a server.>
* <Price> : <Set by "Customer".>
* <Copyright> : <"Customer".>
* <Open Source> : <Code is not open source.>
* <Priority> : <For all interactions with friends a internet connection has to be established, in order to synchronies the action. If not a, action is saved in a file for later synchronization.>
* <Fault tolerance> : <Every action modifying the friends list has to be confirmed at least once.>
* <Response time> : <average time to respond non during gaming is 10s-15s during gaming 1s for selecting max 5s for verification.>
* <Delivery requirement> : <Product delivery according to scheduled time by the course ESE HS 2013.>
* <Ethical requirement> : <No specification about a word filter.>
* <Testability> : <All requirements has to be tested.>
* <Supportability requirement> : <Group 4 ESE HS 2013 is available for support questions and issues.>
* <Disaster recovery> : <System shuts down and no data is saved.>
* <Documentation> : <User manual or short tutorial is available.>
* <Extensibility> : < - >.
* <Usability> : <Specification comes with first prototype.>
* <Accessibility> : <App launched on Swiss stores for android apps, available to all without restrictions.>

Annotation: Tried not to write the same requirements more than one time, some points ar missing requirements but would be covered by redundancy of the other scenarios.

1. Details to be established [↑](#footnote-ref-1)
2. Dictionaries can be managed in the settings screen [↑](#footnote-ref-2)
3. The timer lenght can be set in the settings screen [↑](#footnote-ref-3)
4. Could be used for cheating [↑](#footnote-ref-4)
5. The password is not required for playing once the function is set up. It is a way of identifying yourself when playing on another device than the account is set up on. That way the personal scores and friend database can be ported to a new device. [↑](#footnote-ref-5)
6. To be discussed (exploit possible) [↑](#footnote-ref-6)
7. The entered pattern matches for a word when the pattern is completely contained in the word. However the word has not be exactly equal to the one in the pattern, wich means that the word can have letters in front of and after the pattern. [↑](#footnote-ref-7)
8. A valid name has o start with at least one letter (a-zA-Z) and has to be at least 4 letters long [↑](#footnote-ref-8)
9. Details to be declared [↑](#footnote-ref-9)
10. Details tob e declared [↑](#footnote-ref-10)