Project Requirements

Project Name: Word Finder

Team: 3

Customer: Andrei Chis

Revision History

Version	Date	Revision Description
0.1	02.08.2013	First version for first meeting with customer
0.2	09.08.2013	Use case adaption according to customer input
0.3	16.08.2013	Adapted use cases about 'internet connectivity'. Replaced 'First App Usage' by 'Start Sharing'.
0.4		
0.5		
0.6		

Date: October 16, 2013

Outline

1. Introduction
    1.1 Purpose
    1.2 Stakeholders
    1.3 Definitions
    1.4 System overview
2 Overall Description
    2.1 Use Cases
        Play Game
        Start Sharing
        View Statistics
        Share Game
        Manage Friends
        Manage wordlists
        Save Game
        Share Stats
    2.2 Actor characteristics
3 Specific Requirements
    3.1 Functional Requirements
    3.2 Non-Functional Requirements

1. Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of a Word Finder Game. It will explain the purpose and features of the software system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external input.

1.2 Stakeholders

- Our customer Andrei Chis
- The users

1.3 Definitions

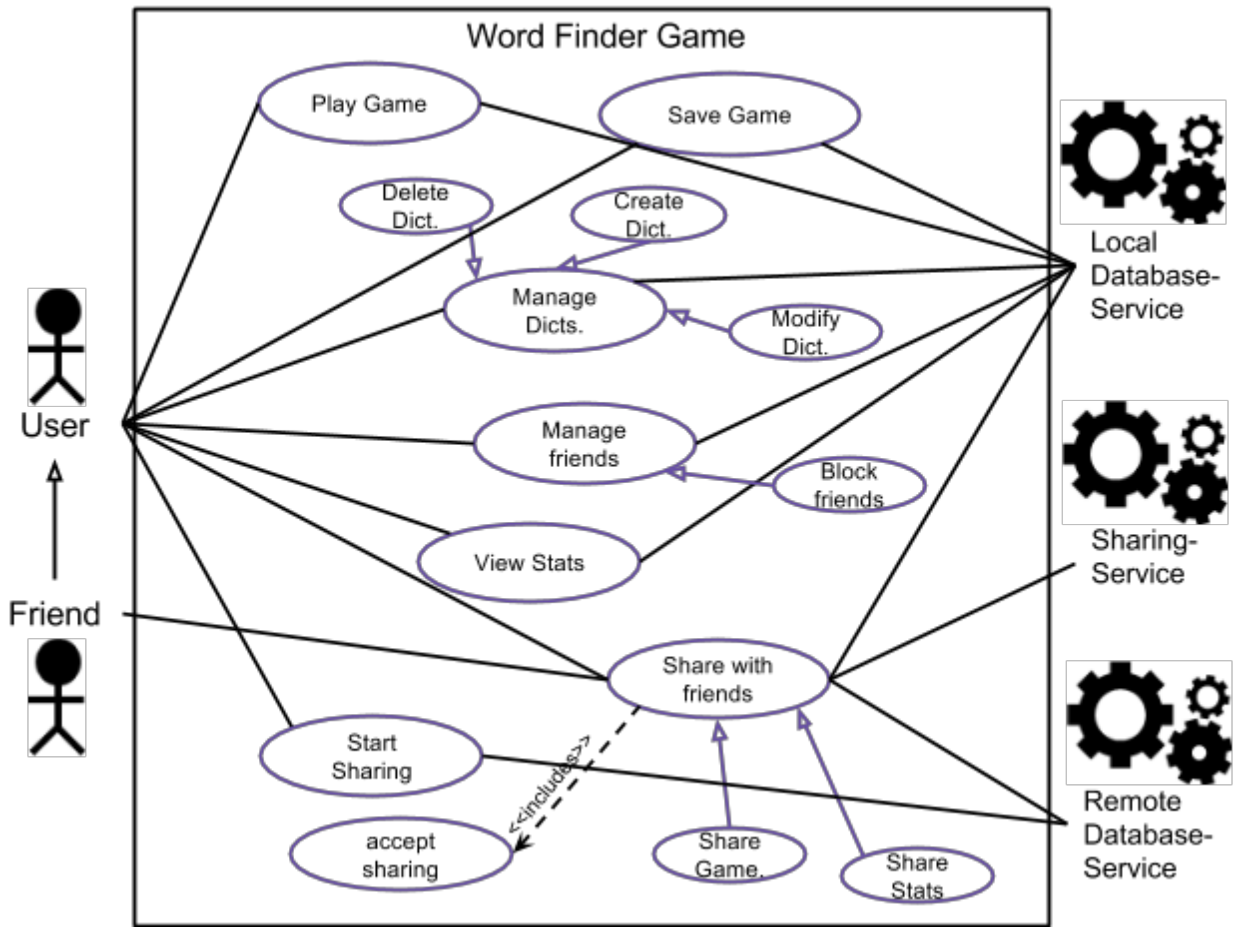
- *Wordlist* - A list of words (one of the user or the default)
- *Board* - A 6x6-matrix where each cell contains a capital letter and a score for this letter.
- *Matrix* - A rectangular alignment of letters with corresponding scores.
- *Friend* - A contact for social interaction.
- *Android* - An operating system for mobile devices.
- *Word* - A sequence of letters. A matching word is a sequence of letters contained in the wordlist.
- *Game* - A combination of a wordlist, board.
- *All-time stats* - The statistics of the user where he can see his best game, his last game and a list of the words he made the most points with.

1.4 System overview

The program is thought to be a simple game which entertains the user and is social too. The user has to find as many words as he can in a given matrix and amount of time and should be able to choose to play a new game or play on the same board on which his friends have played before. In this second case, his score will be ranked in comparison with his friends.
Beside the gaming part of the program, the user should be able to have a look at his and his friends statistics. There is also a possibility to adjust the conditions of the game by adding, create or change the wordlist which is used for creating the game. Those wordlists can also be sent to a friend. It is also possible to share a whole game (board and corresponding words) with a friend.

2 Overall Description

2.1 Use Cases



Play Game

1. **Actors**
   - 1.1. Local user
   - 1.2. A friend of the user

2. **Description**
   - 2.1. The user wants to play a game. He interacts with the board (wiping, rotating) and guessing words while the time is running

3. **Trigger**
   - 3.1. The user clicks on ‘New Game’ or
   - 3.2. The user selects to play on a board he got from a friend

4. **Pre-conditions**
   - 4.1. A board (from a friend or a saved board) has not been played more than 6 times before (total 7 times maximum)

5. Post-conditions

5.1. The score is up to date in all-time stats
5.2. The timer is zero
5.3. Total words guessed stats are up to date
5.4. If played on a board of a friend, the friend gets notified about the score
5.5. If the user cancels a game, the score does not count and the stats remain the same
5.6. The 'Game finished' screen is shown

6. Main-Scenario

6.1. The user starts a new game
6.2. The user guesses a word
6.3. The word is in the current wordlist and gets counted
6.4. The found word appears in the list below
6.5. The score gets updated
6.6. The game is over if the user finds all words, the time runs out or the user cancels the game

7. Alternative Scenario

7.1. During the game, the user gets a call
7.2. The game pauses (Timer stops)
7.3. User finishes the call and returns to the application
7.4. Timer continues counting down

8. Alternative Scenario

8.1. The user starts the game on a board he got from a friend
8.2. He guesses more words than his friend and gets a higher score
8.3. The friend gets notified about the score

9. Notes

9.1. If the user plays a game multiple times (from a friend or a saved game and max 7 times), the best score of all gets counted
9.2. If the computer generates for example 30 words into the board, the game finishes if the user finds 30 words, even if there happens to be some more by accident (not intended by the computer).

Start Sharing

1. Actors

1.1. Local user
1.2. Local Database-Service
1.3. Remote Database Service

2. Description

2.1. The user uses the 'Share' function for the first time and has to register with E-mail and username.

3. Trigger

3.1. The user clicks on ‘Share Game’, ‘Share Stats’ or ‘Share Wordlist’ for the first time.

4. Pre-conditions

4.1. The user did not use the Share function before
4.2. The user has an E-mail address
4.3. The device has an internet connection

5. Post-conditions

5.1. The username and E-mail address will be saved on the game server and on the device by the Database services

6. Main-Scenario

6.1. The user clicks on a Share button somewhere in the application for the first time.
6.2. He gets asked to input a custom username and a valid E-mail address.
6.3. The user enters the requested information
6.4. The information is correct and the user can proceed to share with a friend.

7. Alternative Scenario

7.1. The user enters a username or E-mail address which already exists
7.2. The user gets prompted to reenter his information

8. Alternative Scenario

8.1. The device has no internet connection and the user tries to share
8.2. The user gets asked to enable the internet.
8.3. The user enables the internet and returns to the app.
8.4. The message goes away and the user can enter his information.

9. Notes

9.1. The username and E-mail address can be edited in the options menu
9.2. The server gets contacted to create a new account. The E-mail address is the identifier, but usernames are unique too

View Statistics

1. Actors

1.1. Local user
1.2. Local Database service
1.3. Remote Database service

2. Description

2.1. The user wants to see his statistics and the stats from his friends.

3. Trigger

3.1. The user clicks the “Statistics” Button in the menu.

4. Pre-conditions

4.1. The main menu is shown
4.2. The user has finished at least one board correctly or one of a friend

5. Post-conditions
5.1. The stats remain the same.

6. Main-Scenario
6.1. The user clicks on the “Statistics” button in the menu
6.2. The user sees his all-time stats
6.3. The user sees the last 10 games and the according scores
6.4. The user selects ‘Highscores’ and can see the ten best boards with his highest scores

Share Game

1. Actors
1.1. Local user
1.2. A friend of a user
1.3. Sharing service

2. Description
2.1. The user wants to share a game with one of his friends.

3. Trigger
3.1. The user chooses to share the game from the menu or after a game is finished.

4. Pre-conditions
4.1. The game exists in the user’s pool of different games or the user just finished a game.

5. Post-conditions
5.1. The game is sent to the friend
5.2. The friend will get notified and can accept.
5.3. The saved game remains in the actual state.
5.4. If an internet connection is not present, the app will send the game the next time the internet connection is available.

6. Main-Scenario
6.1. The user clicks the “Share Game” button in the menu.
6.2. The user chooses a game from his pool of saved games.
6.3. The user clicks the “Recipient” button.
6.4. The user chooses a friend from his list of contacts.
6.5. The user clicks the “Send” button to send the game to his friend.
6.6. The friend accepts the game. It will be automatically saved.
6.7. The friend plays the game and his score will be sent back.

7. Alternative Scenario

7.1. After finishing a game the user clicks the “Share Game” button.
7.2. The user clicks the “Recipient” button.
7.3. The user chooses a friend from his list of contacts.
7.4. The user clicks the “Send” button to send the game to his friend.
7.5. The friend accepts the game.
7.6. The friend plays the game and the user can see the score of his friend.

8. Special Requirements

1.1. If the user has not registered before and this is his first time he wants to share something, he has to register according to the case ‘Start Sharing’.

9. Notes

9.1. If no internet connection is available at the time, the user gets promted that the game cannot be shared right now but will be as soon as the internet connection is available again. The Sharing service will take care of this.

Manage Friends

1. Actors

1.1. Local user
1.2. Remote Database Service
1.3. Local Database Service

2. Description

2.1. The user can interact with the Friends-menu to add/delete/block or edit a contact

3. Trigger

3.1. The users clicks the “Friends” button in the main menu to get to the friends-menu. In the friends-menu the user clicks the “Add Friend”, “Delete Friend”, ”Block Friend” or “Edit Friend” button, respectively, depending which task he wants to perform.

4. Pre-conditions

4.1. The application is started successfully and the main menu is displayed.

5. Post-conditions

5.1. Add Friend: A new contact is added to the contact list, while the other contacts remain unchanged.
5.2. Delete Friend: The contact is deleted from the contact list and the other contacts remain unchanged.
5.3. Edit Friend: The contact information is updated while the other contacts remain unchanged, the E-mail of the friend does not change.
5.4. Block Friend: The user does not get notifications about this friend anymore

6. Main-Scenario

6.1. The user clicks the “Add Friend” button.
He enters a name and an E-mail address.

The user clicks the “Save” button to save the newly created contact.

6.2. The user long-presses on a friend in his list.
In the pop-up menu, he selects ‘Delete Friend’.
The friend gets deleted.

6.3. The user long-presses on a friend in his list of friends.
The user selects ‘Edit’ in the pop-up menu.
A new screen appears where the modification can be made.
After clicking on ‘Save’, all modifications get saved.

Special Requirements

6.4. A friend needs a name and an E-mail address.

6.5. The user is not allowed to change the E-mail address of the friend

Manage wordlists

1. Actors

1.1. Local user

1.2. Local Database service

2. Description

2.1. The user creates, modifies, imports and deletes his custom wordlists.

3. Trigger

3.1. The user selects the “Edit wordlists” entry in the preferences menu.

4. Pre-conditions

4.1. Every wordlist must have at least 1 word.

4.2. The user is in the preferences and selects “Edit wordlists”.

4.3. Default wordlists and wordlists which are connected with a shared board can not be modified

5. Post-conditions

5.1. The wordlist list is modified.

5.2. The modifications are saved.

6. Main-Scenario

6.1. The user selects a wordlist to modify.

6.2. The user imports wordlists.

6.3. The user selects “add words” or “import words”.

6.4. The user types in new words or imports them from other wordlists.

6.5. The user saves the changes by selecting the “save changes” button.

7. Alternative Scenario “create wordlist”

7.1. The user selects “create wordlist”.

7.2. The user types a name for his new wordlist.

7.3. The user proceeds as described in the Main-Scenario to add or import new words

Alternative Scenario “delete wordlist”

7.4. The user selects a wordlist he wants to delete

7.5. The user selects “delete this wordlist”
7.6. The user gets asked if he really wants to delete this wordlist
7.7. The user selects “delete”
7.8. The user gets a warning if the wordlist is connected with a saved board (friend or personal).

8. Special Requirements

8.1. All created wordlists need to contain at least 20 words to be used to play (wordlists with less words can be saved but not used to play)

9. Notes

9.1. The default wordlists cannot be modified but used to create new ones.

Save Game

1. Actors

1.1. Local user
1.2. Local Database service

2. Description

2.1. The user can click a ‘Save Game’ button at the end of a game to save the board he played on (plus stats)

3. Trigger

3.1. The user clicks ‘Save Game’

4. Pre-conditions

4.1. The user has successfully finished the game

5. Post-conditions

5.1. The board and the according stats will be saved and appear in the ‘Saved Games’ menu

6. Main-Scenario

6.1. The user wants to save the game he just played to play it again later
6.2. After clicking ‘Save Game’, the user gets prompted to enter a name
6.3. The user enters ‘Very hard words’ so he remembers this board
6.4. Clicking OK saves the game

Share Stats

2. Actors

2.1. Local user
2.2. Sharing service

3. Description

3.1. The user can send his all-time statistics to a friend from his friend-list

4. Trigger

4.1. The user clicks ‘Share Stats’ in the statistics-menu

5. Pre-conditions

5.1. The user has friends in his friend-list
5.2. The user has statistics to share already

6. Post-conditions

6.1. The Stats will be sent to a friend
6.2. The Stats have to be readable for a friend

7. Main-Scenario

7.1. The user chooses a friend from a list
7.2. The user clicks ‘Send’ to submit the Stats
7.3. The user gets a message for the successful submission.

8. Alternative Scenario

8.1. The user chooses a friend from a list
8.2. The user clicks ‘Send’ to submit the stats
8.3. There is no internet connection available or there are some connection issues
8.4. The user gets prompted that the stats will be sent automatically as soon as the internet connection is back

9. Special Requirements

9.1. If the user has not registered before and this is his first time he wants to share something, he has to register according to the case ‘Start Sharing’.

2.2 Actor characteristics

User

The user does not want to get annoyed by large registration forms at the start of the game or pop-ups about connection problems. He just wants to play a game quick and simple. If gets better at the game, he may want to share one or the other board with his friends and then he is willing to register for an account. Also, he does not like to click through endless screens to reach his goal, he likes it simple and easy.

Sharing service

The sharing service is responsible to stack up games, wordlists and statistics if they cannot be shared at the time due to connection issues. He will periodically try to submit the shared content. The sharing service is also responsible for notifications, thus notifies the user about new games he can accept from friends, new friend requests, friend score updates etc. The Sharing service collaborates with the database services.

Local Database service

This service is responsible for the storage of all the games. This means games that were saved manually by the user and also games that were played recently. Also, this service keeps track of the current statistics the user has, the friends he added and their games

he received.

Remote Database service
This remote service operates on the server. He is responsible for the account management and the sharing process between friends. If a user is not only at the time, this service will keep the notifications and shared items until he gets back online.

3 Specific Requirements

3.1 Functional Requirements

1. **In-Game functionalities and requirements**
   - 1.1. While playing, the board can be rotated clockwise
   - 1.2. The score gets updated according to the amount of points given by the word found. Each letter has a specific amount of points based on the frequency in the current wordlist.
   - 1.3. Words can be selected by wiping over them horizontally, vertically and diagonally but not backwards over the same letters
   - 1.4. The game displays how many words were found
   - 1.5. The game displays a timer starting at 5 minutes counting down
   - 1.6. The game cannot be paused except for calls and system notifications
   - 1.7. Words don't count multiple times
   - 1.8. The board contains at least 20 words from the wordlist
2. **Sharing**
   - 2.1. Friends can be added by their E-mail address. The friend has to confirm the request before content can be shared with him.
   - 2.2. At the end of a game, the stats of the game plus the board played on can be shared with friends
   - 2.3. Also, all-time stats can be shared with friends
   - 2.4. Wordlists can be shared
   - 2.5. Offline sharing: The content will be submitted the next time the user has a connection to the internet.
3. **Wordlist management**
   - 3.1. Wordlists can be added and used in game
   - 3.2. Wordlists can be created based on other wordlists and additional words
   - 3.3. Existing wordlists can be edited (adding and deleting words)
   - 3.4. Wordlists connected with saved boards cannot be edited.
   - 3.5. The game comes with default wordlists which cannot be edited
   - 3.6. A wordlist must have at least 20 words
4. **Statistics**
   - 4.1. The user can see his all-time stats (total games played, total words found, total points etc.
   - 4.2. Current stats from friends are shown beside yours
   - 4.3. Statistics can be shared with friends.
5. **Options**
   - 5.1. The user can change its username and E-mail
   - 5.2. Default wordlist selection

3.2 Non-Functional Requirements

1. **Product Requirements**
   - 1.1. The product runs on android devices with API 8 or higher
   - 1.2. The system provides social features (Sharing via E-mail)
   - 1.3. The system does require a server for sharing functionality.
2. **Performance**

2.1. Generating a new board should not exceed two seconds
2.2. The application should be stable (The application does not crash in 100 years)