Project Requirements

Project Name: Word Finder

Team: 3

Customer: Andrei Chis

Revision History

Version	Date	Revision Description
0.1	02.10.2013	First version for first meeting with customer
0.2	09.10.2013	Use case adaption according to customer input
0.3	16.10.2013	Adapted use cases about 'internet connectivity'. Replaced 'First App Usage' by 'Start Sharing'.
0.4	12.11.2013	Tuning for V1 release: Adapted use cases terminology to game and updated purpose
0.5	15.11.2013	Adapt to customers wishes
0.6	27.11.2013	Use cases adapted
1.0	10.12.2013	Final Version

Date: October 16, 2013

Outline

1. Introduction

1.1 Purpose

The Wordfinder game is an app played on android devices challenging the players ability to quickly find words on a 6x6 board containing 36 letters.
The goal of the game is to find as many words as possible up to a maximum to 30 words and try to maximize the score.
Additionally the app adds a social component by offering the user the ability to share his boards, wordlists and his game statistics with friends.

1.2 Stakeholders

- Our customer Andrei Chis
- The users

1.3 Definitions

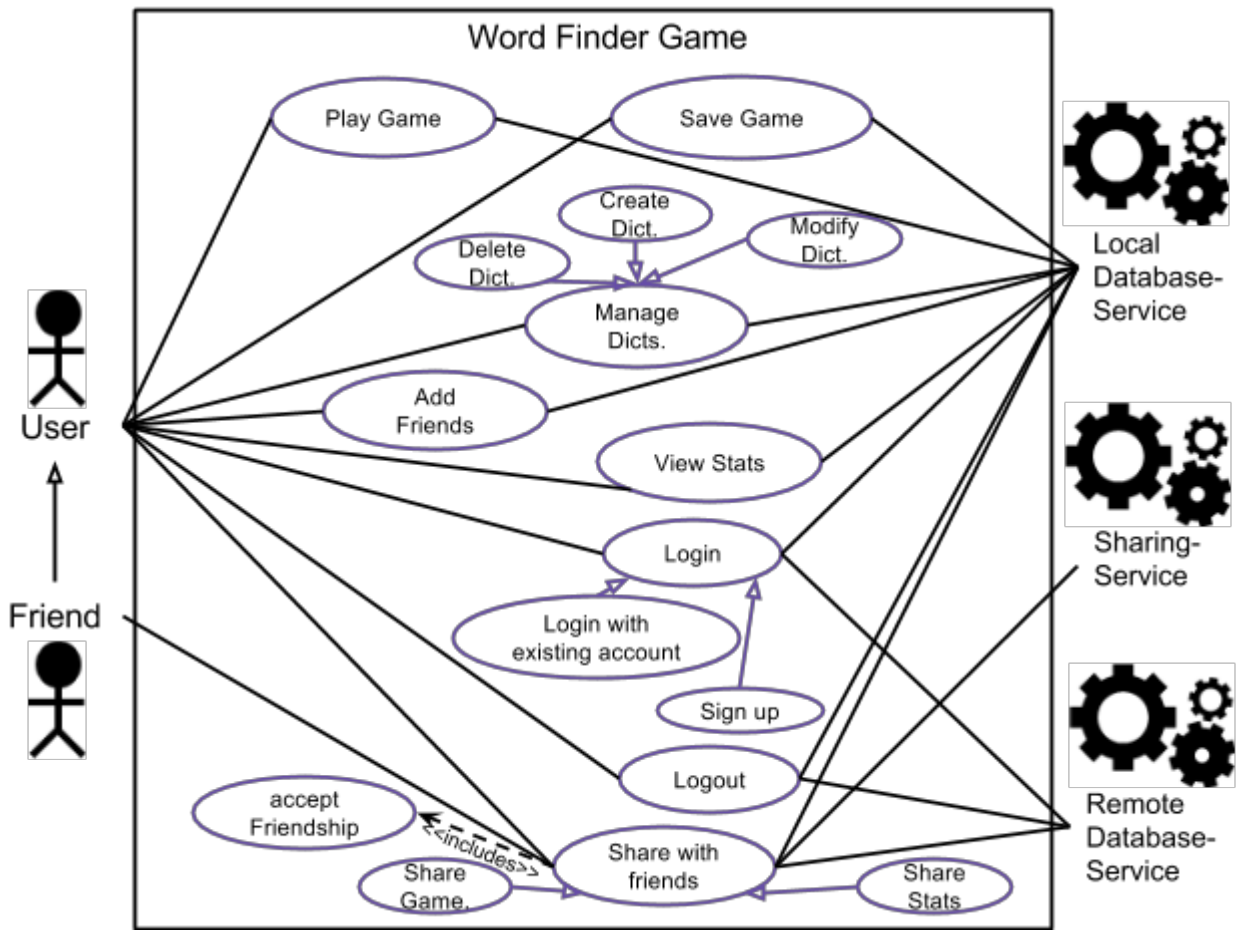
- *Home Screen - The first screen appearing when the app get started*
- *Wordlist* - A list of words (one of the user or the default)
- *Board* - A 6x6-matrix where each cell contains a capital letter and a score for this letter.
- *Matrix* - A rectangular alignment of letters with corresponding scores.
- *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 ‘Start Game’ in the Home Screen, ‘Replay’ at the end of the Game or ‘Replay’ at a selected saved Game.

4. **Preconditions**

5. Postconditions

5.1. The score is up to date in all-time stats
5.2. The timer is zero or the maximum number of words was found (30)
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 over’ 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 or the time runs out
6.7. The user will see the game over screen, which shows reached score, found words, attempted words and elapsed time

7. Alternative Scenario (Pause)

7.1. During the game, the game session gets paused by either pressing the pause button, an incoming call or when user leaves the app without closing it
7.2. The game pauses and the timer stops.
7.3. If the button ‘Pause’ was pressed a pause screen appears
7.4. User presses ‘Resume’ on the pause screen or comes back to the app
7.5. Game screen is shown, the timer continues counting down, user can continue playing

8. Alternative Scenario (Quitting)

8.1. During the game or pause, the user presses quit
8.2. The game is left without any additional options
8.3. The Home Screen appears

9. Alternative Scenario (Replay)

9.1. The user starts the game on a board he got from a friend or he saved or played before.
9.2. On the game over screen he will as addition to the results to the just played game the results of the saved game.
9.3. He guesses more words than his friend or as he had before and gets a higher score
9.4. The friend gets notified about the score
9.5. The user can save or update the game

10. Notes

10.1. 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).

Share with friends

1. Actors
- 1.1. Local user
- 1.2. (Friend)
- 1.3. Local Database-Service
- 1.4. Remote Database Service

2. Description
- 2.1. The user uses the ‘Share’ function to share games, statistics or wordlist.

3. Trigger
- 3.1. The user clicks on ‘Share Game’, ‘Share Stats’ or ‘Share Wordlist’

4. Preconditions
- 4.1. The user is logged in
- 4.2. The user have friends on the application
- 4.3. The device has an internet connection

5. Postconditions
- 5.1. The object to share 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.
- 6.2. He gets asked to select a friend to share with.

7. Alternative Scenario
- 7.1. The user is not logged in
- 7.2. The user gets prompted to login first
- 7.3. After the login the user can continue sharing.

8. Alternative Scenario
- 8.1. The user does not have friends on the application
- 8.2. The user informed to create first some friends before
- 8.3. After adding some friends, the user can choose them, when asked to select.
- 8.4. The user can continue sharing.

9. Alternative Scenario
- 9.1. The device has no internet connection and the user tries to share
- 9.2. The user gets asked to enable the internet.
- 9.3. The user enables the internet and returns to the app.
- 9.4. The message goes away and the user can continue sharing.

10. Notes
- 10.1. If the user has no account to login with, he has first to create a new account.

Login

1. Actors

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

2. Description

2.1. The user can login to use the ‘Share’ function and manage his freinds. The login require as input a valid account with username and password.

3. Trigger

3.1. The user clicks on ‘Preferences‘ -> ‘Login with existing Account’

4. Preconditions

4.1. The user is not logged in yet
4.2. The user has created an account
4.3. The device has an internet connection

5. Postconditions

5.1. The use is logged in
5.2. The user can use the ‘Share’ function and friend management.

6. Main-Scenario

6.1. The user clicks on ‘Preferences‘ -> ‘Login with existing Account’.
6.2. He gets asked to input a custom username and password.
6.3. The user enters the requested information
6.4. The information is correct and the user can proceed using different other function such as ‘Share’ and others.

7. Alternative Scenario

7.1. The user enters a username or password which does not 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 login
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. After the login, a user can log out by pressing ‘Preferences‘ -> ‘Logout’. After the logout the user is not any longer logged in and can not use the special functions until another login action.

Create new Account

1. Actors

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

2. Description

2.1. The user creates a new account for possible login action. The user has to register with username, password and E-mail.

3. Trigger

3.1. The user clicks on ‘Preferences‘ -> ‘Create new Account’

4. Preconditions

4.1. The user has an E-mail address
4.2. The device has an internet connection

5. Postconditions

5.1. The username and E-mail address will be saved on the game server and on the device by the Database services
5.2. The user can login with the created account

6. Main-Scenario

6.1. The user clicks on ‘Preferences‘ -> ‘Create new Account’
6.2. He gets asked to input a custom username, a password and a valid E-mail address.
6.3. The user enters the requested information
6.4. The information are complete and this new account will be saved on the game server

7. Alternative Scenario

7.1. The user enters a username or E-mail address which already exists or is not valid
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 server gets contacted to create a new account. The E-mail address is the identifier, but usernames are unique too

Add friends

1. Actors

1.1. Local user
1.2. (Friend)
1.3. Local Database-Service
1.4. Remote Database Service

2. Description

2.1. The user can send a friend request to a friend, who can accept or reject. By accepting, the friendship is added on the game server and on the devices.

3. Trigger

3.1. The user clicks on ‘Add Friend’ on the ‘Friends’-screen and put a email of a friend in.
3.2. The friend accepts the request.

4. Preconditions

4.1. The user is logged in
4.2. The friend is saved as user on the game server (the user has this app and an account on it)
4.3. The friend will be logged in once in the future
4.4. The device has an internet connection

5. Postconditions

5.1. The friendship with both userIDs 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 ‘Add Friend’ on the ‘Friends’-screen.
6.2. He gets asked to input a friend’s valid E-mail address.
6.3. The user enters the requested information
6.4. The information is correct and a request is saved on the server.
6.5. The friend let display his requests.
6.6. The friend accepts the request.
6.7. The request is deleted and the friendship will be saved on the game server and on the device by the Database services.

7. Alternative Scenario

7.1. The device has no internet connection and the user tries to send a request
7.2. The user gets asked to enable the internet.
7.3. The user enables the internet and returns to the app.
7.4. The message goes away and the user can continue.

8. Alternative Scenario

8.1. The user or the friend deletes the request.
8.2. The request is deleted.
8.3. No friend will be added.

9. Alternative Scenario

9.1. The friend does not answer the request
9.2. Nothing will happen; request will stay there and no user is added.

10. Notes

10.1. A friend can be deleted by long-pressing on the friend and pressing delete. The friend will be removed but can be added again.

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. Preconditions

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

5. Postconditions

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. Preconditions

4.1. The game exists in the user's pool of different games or the user just finished a game.
4.2. The user started sharing

5. Postconditions

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. Preconditions

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

5. Postconditions

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.

7. Special Requirements

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

7.2. 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. Preconditions

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. Postconditions

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’ button at the end of a game to save the board he played on (plus stats)

3. Trigger
3.1. The user clicks ‘Save’ after the game was finished

4. Preconditions
4.1. The user has successfully finished the game

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

6. Postconditions
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)