+JMJ+

Software Requirements Specification

for

Oxword Fictionary

Version 1.1 approved

Prepared by Ben Campbell

The Fictionary Team

February 17, 2021

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
	2/11/2021	First version	1.0
	2/27/2021	Style update to GitHub Markdown	1.1

Introduction

Purpose

This document specifies the requirements for the application Oxword Fictionary, version 1.0. This version covers both Android and iOS applications for phones.

Document Conventions

There are three levels of requirements:

1. **Essential:** these are the basic requirements that the app must have in order to do what it does.

2. **Recommended:** these are not essential, but still very useful and needed in order to implement some of the optional requirements

3. **Optional:** these are non-essential and the application may function perfectly well without them, but they make it more interesting, engaging and fun.

Intended Audience and Reading Suggestions

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

TBD.

Quick Description

Oxword Fictionary is a fake dictionary phone app for entertainment and creativity. It was built off of the idea that people sometimes use strange and fun words for situations even

though the words aren't real words. This dictionary gives them the opportunity to **make** them real words!

One of the basic features is creating fake words, and discovering other words that people made up. Creativity is encouraged, and the app will be designed in order to make things as fun and smooth as possible. Things that will help with this are ways to upvote words, add tags and comments to them, multiple definitions, and possibly several creativity games that present people with words and challenge them to make definitions for it, or vise-versa.

Since this app is being created by college students for a class project, we don't really have any business goals or strategies except to have a fun time, learn a lot, and get a good grade in the end.

References

No specification documents or style guides were used in the creation of this plan, although there are a few links to the Microsoft documentation website, as well as the project GitHub Page:

General

Microsoft Documentation Website : https://docs.microsoft.com/en-us/

Tools

Visual Studio : https://visualstudio.microsoft.com/
Xamarin : https://docs.microsoft.com/en-us/xamarin/get-started/what-is-xamarin
Xamarin Forms : https://docs.microsoft.com/en-us/xamarin/xamarin-forms/
DreamHost : https://www.dreamhost.com/
MySQL : https://www.mysql.com/
GitHub : https://github.com/
Git : https://git-scm.com/

The Fictionary Team

Oxword Fictionary GitHub Page : https://github.com/BenTBCampbell/Software-Engineering
GitHub Wiki: https://github.com/BenTBCampbell/Software-Engineering/wiki

Overall Description

Product Perspective

Oxword Fictionary is a brand-new, self-contained product. It is not an extension or modification of any other application, although it does have two sub-applications: the Android version, and the iOS version. Most of the code will be shared between both versions, but there will be a little bit platform-specific code for specific details. The Microsoft documentation page What is Xamarin? can give a better idea of how this works.

Product Functions

The main functions of the app are:

Add new fake words to the dictionary

Being able to find words other people made

More details are covered in section 4.

User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

TBD.

Operating Environment

Oxword Fictionary will be available for iOS and Android phones. Details are TBD.

Design and Implementation Constraints

Oxword Fictionary will be available in English, and English only.

The Fictionary Team currently has no corporate policies whatsoever, but we are under the policies of Franciscan University of Steubenville, as well as Professor Wessel, the teacher of our Software Engineering class, who will decide the final grade of this project.

Here are some of the software that we are using. The limitations caused by these are currently unknown:

Database: MySQL, hosted by DreamHost

Software: C# programmed in Xamarin in VisualStudio

Hardware: Android and iOS phones

It is unlikely that this application will continue to be maintained after the completion of the project.

Security measures and programming standards are TBD.

User Documentation

User documentation is planned to be on the GitHub wiki. There are no current plans to make tutorials, although they may be made in the future. Documentation and delivery standards are TBD.

Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you

intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

See the list in section 2.5 to see what software and tools are being used. Issues and constraints about these tools are currently unknown.

External Interface Requirements

User Interfaces

We will be programming our interface with Xamarin Forms. details are TBD, but here is a list of the essential pages and elements:

- **Home Page**: Displays the word of the day, maybe a few fun facts about things, new and popular words
- **Search bar:** a bar to search for words by full or partial words, usernames, tags, or possibly date of creation
- **Search page**: a page to display search results
- **Create word page:** a page to create and submit new words
- **Word entry pages:** A page to display information about a specific word, such as its definition, tags, comments, and other things.
- **Login page:** a page to log in
- **Create account page:** a page to create an account
- **User account pages:** pages for particular users and what words they made, or an option to modify your own account settings
- **Creativity game pages:** Some pages with games and activities to help encourage creativity in order to make new words
- **Settings page:** a page to manage settings like the color theme and account information
- **Credits page:** a page to give a shout-out for all the contributions people made

Hardware Interfaces

The only hardware components are the smart phones the users own, and the communication and protocols between our software and the hardware will be managed by

the operating system, which is outside the scope of this project. All we need to consider are the APIs made available to us through Xamarin.

Software Interfaces

Oxword Fictionary will be interacting with a MySQL Database (version 5.7.28) in order to save information for user accounts and words. The details on how this is going to work is TBD

Communications Interfaces

Oxword Fictionary does not currently have any communication functions besides communicating with the database. This is TBD, so we don't know the details of security or encryption issues. Since the app is not expected to be very popular, there will not likely be issues with too much traffic.

One possible issue with synchronization is that there needs to be a way for the user's local app to know when a new word is added. A proper solution for this is TBD, but it may not end up being a significant problem.

System Features

Essential Features

These are the most basic and essential features of the application. Without them, the application would not be what it is supposed to be. They must be implemented before any of the other features are.

Inputting Words

Description

The users must be able to submit new word entries into the fake dictionary.

Stimulus/Response Sequences

1. Go to submit new word screen
2. Type in word and definition
3. Optionally attach tags to it
4. Press submit, word and definition are screened, and sent to the database if successful

Functional Requirements (REQs):

1. Background dictionary in order to determine if the current entry is a real word
2. A way to check if the word already has an entry in the fake dictionary
3. A filter (REQ 4.1.2) to make sure there are no profanities in the word or the definition
4. A way to attach data to entries such as which user created it, what tags

Word Input Filter

Description

There must be a filter so users can't submit anything inappropriate in new words

Functional Requirements

1. A way to check for regular bad words
2. A way to check for hidden or misspelled but still obvious bad words

Searching for Words

Description

The users must be able to find other words in the fake dictionary

Stimulus/Response Sequences

1. Type in search bar or go to search screen
2. Type in word, part of a word, a tag, or a user name
3. The App queries the database and displays results that match the search

Functional Requirements

A way to query the database by:

1. full words
2. parts of words
3. user names
4. tags
5. when the word was made
6. how popular the word is

A way to display most relevant searches in a search menu and/or a search page in an intuitive way, including:

1. word name
2. definition
3. tags
4. possibly the ability to upvote

Word Entry Page

Description

There must be some sort of screen to display a word, its definition, and other information about the word such as comments, tags, and ratings.

Stimulus/Response Sequences

1. Choose a word from the search screen
2. A new screen shows up that displays information about the word

Functional Requirements

A way to query the database for information about a specific word:

1. definitions
2. tags
3. user who created it

4. comments
5. ratings

A way to display this information, as well as access other features related to the word:

1. Upvoting
2. Comments box

Recommended Features

These are the features that are not absolutely essential for the app to do what it's supposed to, but they are definitely recommended and foundational for a lot of the optional features.

User Accounts

Description

The users will have the opportunity to make accounts so that they can claim credit for submitting words. The details are TBD.

Stimulus/Response Sequences

1. User goes to create account page
2. The user makes an account

Functional Requirements

1. Creating an account.

Optional Features

These are the features that are not needed for the application to do what it is supposed to, but they add color and flavor, and are what will make the app fun and entertaining in the end. If some of these are not included in the final product, it is not a big deal.

Word of the Day

Description

There will be a word of the day on the home screen to show off interesting or highly-rated words

Stimulus/Response Sequences

1. User sees the word of the day on the home screen
2. If the user pushes it, they go to the entry page for that word

Functional Requirements

1. Choosing a word of the day based on factors that are TBD
2. Displaying the word in a nice way on the home page

Upvoting words

Description

The users will have the ability to upvote a word if they like it

Stimulus/Response Sequences

1. User upvotes a word from it's entry page
2. User upvotes a word directly from the search page or search menu

Functional Requirements

1. having a way to upvote a word from its entry page
2. having a way to upvote a word from its search page
3. having a way to upvote a word from its entry in the search bar
4. having a way to send information about an upvote to the database
5. preventing a user from being able to upvote a word multiple times

Word Comments

Description

The users should be able to make appropriate comments on a word if they choose

Stimulus/Response Sequences

1. User goes to the word entry page
2. The user types a comment in the comment box
3. The user pushes the submit button
4. The comment is screened to make sure it does not have any inappropriate content
5. The comment is sent to the database

Functional Requirements

1. An interface to submit comments
2. A way to screen comments for inappropriate content
3. A way to submit comments about a word to the database
4. A way to flag bad or cruel comments in case they get past the filter
5. An easy way for an administrator to review and remove/resolve flagged comments

Word Tags

Description

The users will have the ability to add tags/categories to words when they make them, as well as search for words by their tags

Stimulus/Response Sequences

1. User submits a tag when they create the word
2. Other users can search for words by the tag

Functional Requirements

1. A way to add tags on the submit new entry page
2. A way to see what tags already exist so redundant ones are not created
3. A way to search by tags

4. Possibly a way to group related tags together

Display Themes

Description

The users should be able to choose how general app interface and colors look. There should at least be a light and dark theme to match system preferences

Stimulus/Response Sequences

1. User goes to settings page
2. User chooses theme

Functional Requirements

1. A way to store what the current theme is on the local device
2. A way to quickly change the colors, font, and possibly icons of the app based on the theme
3. A way to set the theme to be whatever the system default is

Other Nonfunctional Requirements

Performance Requirements

Oxword Fictionary is not expected to be a very resource-intensive app, so there are no specific performance requirements currently planned. Some things to consider are basic response-times, such as how long it takes for a screen to load, or for a button press to cause a reaction. Since not many details are known about the specific screens of the app yet, these are TBD.

Safety Requirements

There are no expected ways in which users could experience loss, damage, or harm, except through mean, cruel, or abusive words or comments. This kind of harm is hoped to be avoided through the use of filters.

Security Requirements

This app is not expected to contain much personal information. A lot of that depends on how we implement accounts, which is TBD, so most of safety and security is also TBD at this time.

Software Quality Attributes

This is expected to be a relatively simple app, so priority will be placed over ease of use. It won't take much effort to learn.

Specifically this means that we will have to focus on how to make our menus intuitive, placing things in places where people will expect to find them, making them look nice through proper fonts, text sizes, and colors, having nice icons and menus, and designing things so that it is possible to accomplish things like searching, commenting, and submitting new words without many actions or effort.

Business Roles

There are not many roles planned, but at the very least there will be users who will have access to all the basic features of the app, and developers who are able to modify the application and database, as well as do whatever users are able to do.

If comments are implemented, there will need to be admins to monitor them and resolve/remove flagged comments. These will probably just be the developers.

Other Requirements

TBD

Appendix A: Glossary

Terms

Oxword Fictionary – The fake dictionary app we're making, silly!
The Fictionary Team – The people developing it. Namely Ben, Catherine, Christopher, and Margs
Xamarin: A platform for making apps in C# for several devices that share a lot of code between them
Xamarin Forms: The UI interface for Xamarin
Visual Studio: The Microsoft IDE we will be using to program our app.

Abbreviations

IDE – Integrated Development Enviornment
TBD – To Be Determined
REQ – Requirement

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

TBD

Appendix C: To Be Determined List

Requirement	Section
1 Introduction	
Intended audience and reading suggestions	1.3
2 Overall Description	
User Classes and characteristics	2.3