Software Requirements Specification

for

NO Monkeying Around!

Version 1.0 approved

Prepared by D. Brinn & S. Staton

Android Monke

17 February 2022

Table of Contents

Table of Contents i
Revision History ii
1. Introduction 1
1.1 Purpose
1.2 Document Conventions
1.3 Intended Audience and Reading Suggestions
1.4 Product Scope
1.5 References
2. Overall Description 2
2.1 Product Perspective
2.2 Product Functions
2.3 User Classes and Characteristics
2.4 Operating Environment
2.5 Design and Implementation Constraints
2.6 User Documentation
2.7 Assumptions and Dependencies
3. External Interface Requirements 3
3.1 User Interfaces
3.2 Hardware Interfaces
3.3 Software Interfaces
3.4 Communications Interfaces
4. System Features 5
4.1 System Feature 1
4.2 System Feature 2 (and so on)
5. Other Nonfunctional Requirements 9
5.1 Performance Requirements
5.2 Safety Requirements
5.3 Security Requirements
5.4 Software Quality Attributes
5.5 Business Rules
6. Other Requirements 10
Appendix A: Glossary 10
Appendix B: Analysis Models 10
Appendix C: To Be Determined List 10

Revision History

Name	Date	Reason For Changes	Version
NO Monkeying Around!	2/20/2022	Initial Document Creation	1.0
NO Monkeying Around!	3/6/2022	Document Revision for Deliverable 2	1.1
NO Monkeying Around!	3/24/2022	Document Revision for Deliverable 3	1.2

1. Introduction

1.1 Purpose

This Software Requirements Specification document outlines features and requirements for the android application NO Monkeying Around! version 1.0. This application will be a mobile application available on the Google Play store. This SRS will describe functionalities of the entire system.

1.2 Document Conventions

For this version of the document, no specific document conventions have been used. For the purposes of deliverable 1, sections which do not need to be completed have been grayed out.

1.3 Intended Audience and Reading Suggestions

This document is intended mainly for the development team as well as possible specific user groups such as beta testers. Another potential audience for this document is management staff looking to implement NMA! as part of an educational or business environment. The rest of this SRS document will detail different usable features of NMA! as well as specific privacy and security requirements that will be implemented. It is suggested that users first read section 2, which gives an overall description of the application, then sections concerning the user interface, system features, and finally sections on privacy and security.

1.4 Product Scope

This application is a productivity application which encourages the user to engage with the app and stay productive by having the user grow a tree. The more the user stays on task and completes things on a to-do list, the more the tree grows. This application is beneficial for the user, because it encourages them to form good study/work habits which has the potential to improve grades/professional performance. In a business environment, this could be implemented as part of a company wide calendar system, allowing management to add meetings and project deadlines.

1.5 References

2. Overall Description

2.1 Product Perspective

NMA! is a new, self contained productivity application. It does not expand on any current system, or serve as a replacement for an existing system. It will consist of an android application that communicates with a database to store user information. It will also be able to communicate with other user devices. This

application is developed for users ranging from regular people trying to keep a better handle of their schedules, to teachers wanting to implement the application as a classroom teaching tool.

2.2 Product Functions

Summary of major application functional requirements:

1. The system shall provide a calendar as part of the UI.
2. The system shall make the tree grow if the users complete tasks
3. The system shall make the tree grow if the user spends time on the application.
4. The system shall grant the user point boosts for consecutive use.
5. The system shall let the user add tasks to a to-do list.
6. The system shall notify the user when they have upcoming due dates.
7. The system shall allow the user to unlock new trees the more they use the app.
8. The system shall let the user export data to a separate device.
9. The system shall allow the user to view how many trees they've grown.
10. The system shall allow the user to set time-sensitive goals on the calendar.
11. The system shall allow the user to log-in to the service through existing accounts (i.e Google)
12. The system shall allow the user to log-in as a "guest".
13. The system shall allow the user to input their financial information for in app purchases.
14. The system shall allow the user to purchase boosts to their time-on-task points.
15. The system shall allow the user to purchase "freezes".
16. The system shall allow the user to use "freezes" to keep their
17. The system shall provide the user a global leaderboard to track others' progress.
18. The system shall provide the user additional unlockable trees for points.
19. The system shall provide a terms of service.

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

1. Financial Dependencies

a. Due to the implementation of microtransactions, a third party program to facilitate these is necessary. It is the security of these financials that may pose a risk from the stated non-functional requirements.

2. Server Dependencies

a. Due to the implementation of a global leaderboard, which necessitates a central database of user information, a third party program to facilitate this may be necessary. The upkeep of this database may pose a risk of shortcoming to the stated non-functional requirements.

3. External Interface Requirements

3.1 User Interfaces

1. **Initialization Page**: The user will be greeted with a graphic, displaying the name of the application, providing an option to log-in or play as a guest. The log-in integration is displayed here, not in a separate page, if the user chooses to do so.

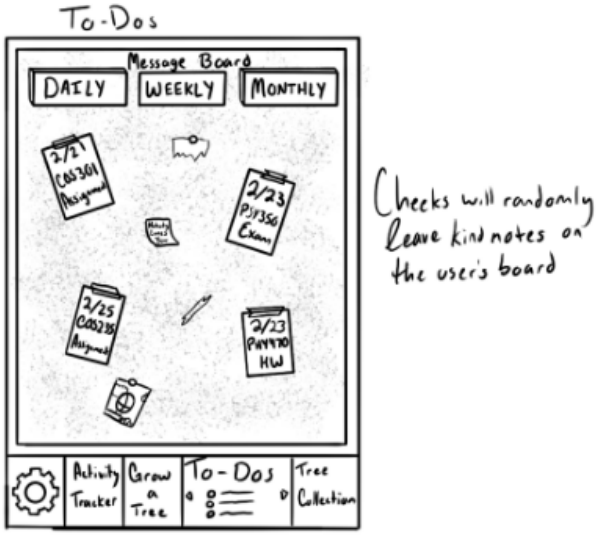
a. **Main Page**: The "main page" displays the growing tree, and an increasing counter of time-on-task score. This is the "idle game" of the application, and it is here where notifications for upcoming tasks will be displayed. The bottom of this screen has a "taskbar", for additional popups.

i. **Store Tab**: A tab on the bottom-right of the taskbar, when clicked, will display a popup to customize one's tree. Here, it is shown that one may trade-in a score for different types of trees, or choose from the trees they have already earned. Boosts and modifiers are locked behind a paywall, and if they are clicked, the page will proceed to a checkout popup.

1. **Checkout Popup**: This simply displays an optional TOS before taking the user through a financial transaction.

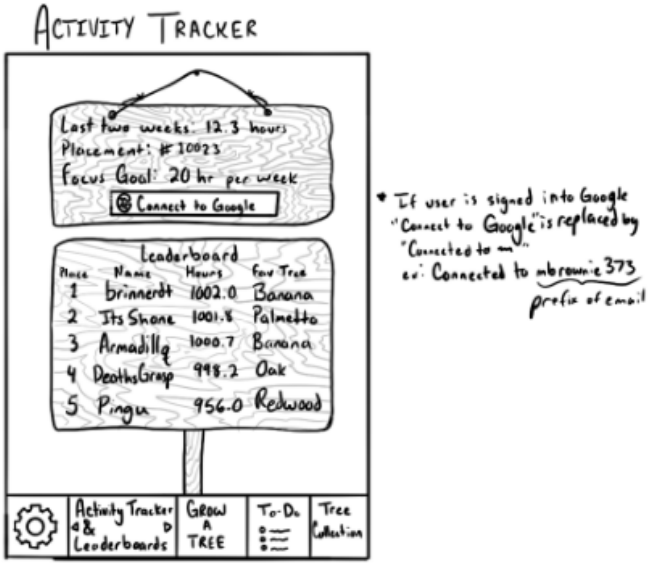
ii. **Calendar Tab**:

1. This tab in the middle of the taskbar, when clicked, opens a popup of a calendar. The user may click different days, initiate a sub-popup, that then prompts them for due date times of day, the name of the work, and a brief description.



iii. **"More" Tab**: This tab sists on the bottom-left of the taskbar, when opened, has the following options;

1. **Settings Popup**: This popup, when initiated, will allow the user to change volume settings, their financial information, their user profile, and so on.
2. **Log-in Popup**: This popup will allow the user to log-in if they had chosen "guest" initially.
3. **Leaderboard Popup**: This popup shows the global leaderboard in descending order.



3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

4. System Features

4.1 REQ-1: Main Page/Animated Tree

4.1.1 Description:

This is an incredibly important feature of NMA! On the main page of the application, an animated tree will be displayed. This tree starts as a sapling and then grows based on the time that the user spends in the app, accounting for any multipliers which are actively being applied. The application will award the user with growth multipliers for completing to-do list items and consecutive use of the application. New trees will be able to be unlocked by the user, the more trees the user grows. This is another high priority feature.

4.1.2 Stimulus/Response Sequences

When the user initially opens the application is when this main page will be displayed. The current tree being grown by the user is displayed. In addition to the tree, the main page will display at the top of the screen the score for the current tree, any multipliers the user has added, and a message from Cheeks (our monkey mascot). From this main page, the user will be able to open the menu tab and navigate to other pages of the application.

4.1.3 Functional Requirements

TBD for this feature.

4.2 REQ-2: Calendar

4.2.1 Description:

As part of our application, we will implement a calendar for the user as part of the UI. This calendar will give the user the ability to add events and view them. The calendar will be able to display events for the day, week or month. Since this functionality will be a major part of the UI, this is a high priority feature. A basic to-do list will also be accessible through this page.

4.2.2 Stimulus/Response Sequences:

In order to navigate to the calendar the user would click on the menu button. The user would then click on the calendar option which would bring them to the page displaying their calendar. On this page the user would have the option to change the calendar view (daily, weekly, monthly) as well as be able to add or remove events.

4.2.3 Functional Requirements:

In order to implement this calendar feature, a database system of some kind will be needed. Google account login integration will also call for additional functional requirements.

4.3 REQ-3: Store Page

4.3.1 Description:

Another feature of the application, is a store page which enables the user to purchase boosts for their tree, different trees, and “freezes” which allow the user to close the application without killing their current tree. The store page is a medium priority feature.

4.3.2 Stimulus/Response Sequences:

To navigate to this page, the user will open the menu from the main page and select the store option. Once the page is displayed, there will be options to purchase boosts and freezes. When the user chooses to buy something from the store, they will be prompted to log into the Google play store. The app will use the Google play store in order to enable users to engage in the in app purchases.

4.3.3 Functional Requirements

In order to implement in-app purchases, the application will need to communicate with the Google play store. In addition to this, sensitive information such as banking information or credit card details will need to be stored. This requires additional security requirements.

4.4 REQ-4: Leaderboard

4.4.1 Description:

As one of the tabs displayed on the main page, users will be able to view a leaderboard. This leaderboard will be able to display rankings of all users on the application. This is a medium priority feature.

4.4.2 Stimulus/Response Sequences:

From the menu displayed on the main page, the user will be able to navigate to the leaderboard by clicking the leaderboard option. This will display the leaderboard.

4.4.3 Functional Requirements:

With the implementation of a leaderboard, a database system of some kind will again be a required piece of the back end environment. The data for each user will need to be stored in a manner that is easy to visualize and pull from as part of the leaderboard functionality. Other additional functional requirements for this app function are TBD

4.5 REQ-5: Leaderboard

4.5.1 Description:

This is a reachable page/tab from the main screen which allows the user to change different aspects of the application. It will allow the user to change things like app themes, enable/disable music on the application, and change certain app permissions. This is a medium/lower priority feature

4.5.2 Stimulus/Response Sequences:

Users will be able to get to this tab of the application through a button on the home screen. Once in the tab, the user will see different settings categories which can then be expanded to show individual changeable settings.

4.5.3 Functional Requirements:

TBD

4.6 REQ-4: Notifications

4.6.1 Description:

Notifications will appear on the user's device alerting them of upcoming due dates/ to-do list items. The user will also be notified when a freeze for a tree is ending. Notifications sent by the application will be able to be disabled or enabled in device settings, similar to apps like facebook or gmail. This is a high priority feature.

4.6.1 Stimulus/Response Sequences:

These notifications will function the same way as notifications for other applications. The application will display the notifications on the user's device, and the user can then click on the notification, opening the app, or swipe it left or right to ignore it.

4.6.3 Functional Requirements:

Since the notification will need to be displayed on the user's device, the software needs to be able to communicate with the device's operating system to display these notifications to the user. Other functional requirements for this feature still need to be determined.

4.7 REQ-7: Data Exporting

4.7.1 Description:

The feature is primarily geared towards implementation of this application within a classroom setting. The user should be able to send their time spent on the application to another device. This would allow the teacher to receive data from their class of students. This is a low priority function.

4.7.2 Stimulus/Response Sequences:

Initially, the teacher would need to link their student's accounts to their own. After this, the student users would be able to press on a button to simply send their weekly time spent data to the teacher's device. The teacher would then be able to view her class data through the leaderboard page in the app.

4.7.3 Functional Requirements:

This is another feature that will require the implementation of a database. Other functional requirements for this feature are TBD.

4.7 REQ-8: Tree Collection

4.7.1 Description:

The user will be able to view a page which displays the last 10 trees that they grew.

4.7.2 Stimulus/Response Sequences:

As one of the tabs on the main page the user will be able to click on the tree collection tab. This will bring them to a page displaying their 10 most recent trees. The user will be able to click on the tree and see the score associated with it.

4.7.3 Functional Requirements:

A database will need to be implemented in the backend to store the user's past tree objects and display them on the application. Other functional requirements are TBD.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

1. The system shall be able to process touch screen input within 3 seconds, 97% of the time.
2. The system shall communicate with a database to store user data.
3. The system shall communicate with a database to query user data.
4. The system shall be able to maintain the server 97% of the time.
5. The system shall be able to maintain the leaderboard 97% of the time.

5.2 Safety Requirements

5.3 Security Requirements

1. The system shall be able to store user's data securely.
2. The system shall adhere to the CCPA.
3. The system shall adhere to the GDPR.
4. The system shall prevent security breaches during financial transactions 98% of the time.

5.4 Software Quality Attributes

5.4.1 Reliability

The system shall be designed in such a way where application crashes happen less than 3% of the time that the user is using the application. The system database shall also be designed in a way where system calls to the database take less than 2 seconds.

5.4.2 Transferability

The system code shall be written in a manner that will transfer data between some fashion of a central server and various user devices - completing 99% of every request.

5.4.3 Useability

The interfaces will be designed in user-friendly, and self-explanatory fashion. Therein, no form of instruction will be needed to navigate and use the software.

5.5 Business Rules

Developer:
The developer role, needless to say, has the privileges of implementing functionalities and features of the application, and is tasked with facilitating the aforementioned functional requirements for the user to engage with.

End-user:
The end-user will be able to use all of the functionalities listed in section 4. The app, itself, does not distinguish between different user roles.

6. Other Requirements

Appendix A: Glossary

Appendix B: Analysis Models

Appendix C: To Be Determined List