Software Requirements Specification
Note-ify

Group 12: Darin Barth, Matt Martin, and Priamwad Poudel

Date	Changes	Version
9/15/17	Initial Document (Sections 1, 2, and 3.2 added)	1.0
11/12/17	Revisions based on Limited Phase 1 Completion; used to create backlog for Phase 5	1.1

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to explain the requirements and features of the note and task manager.

1.2 Scope

The note manager is an all-in-one scheduler, task-manager, and journal, meant to house users' plans and thoughts as well as boost the user's productivity by helping to narrow his focus on what really needs to get done. The note manager is a web application that will be made using the Meteor Javascript Framework. Meteor also includes a database system called MongoDB that will house all our user login information as well as their journal entries.

After the user logs into the website, they will be greeted by a huge calendar. The user can then click on any date to create an entry. An entry can be a myriad of things such as calendar events, meetings with other users, notes, tasks, and projects. The user can also tag an entry if they want to organized a task by any form of classification that can be set up based on a user's preferences. A search system will be implemented so the user can quickly find the notes or tags he needs. Finally, when the user is done with viewing their entries, they can exit with a logout button.

1.3 Definitions, Acronyms, and Abbreviations

1.4 Overview

The rest of the document will go more indepth regarding the overall description and the specific requirements for our application. The overall description will contain the product's perspective and the functions. The specific requirements will cover our system's needs and requirements.

2. Overall Description

2.1 Product Perspective

2.1.1 System Interfaces

The task managing web application will not require additional systems to run on any browser.

2.1.2 User Interfaces

The user interface will consist of navigating through the web application with a mouse and keyboard or if available, through a touchscreen.

2.1.3 Hardware Interfaces

All devices including desktop and mobile devices that can run a modern web browser should be able to load and use the web application

2.1.4 Software Interfaces

This web application can be used via a modern web browser on desktop and mobile devices.

2.1.5 Communications Interfaces

The web application will send and receive information from the database.

2.1.6 Memory

This web application will use very little of the browsers RAM. It does not need any storage besides a cache on the users side of the web application as all saved data is sent to the database for secure storage.

2.1.7 Operations

There are several modes of operations for the web application. The main mode is the task viewer and creator where the user creates tasks. The calendar mode takes any task information that the user has entered and displays it in a calendar view. There is also a search mode that goes through and finds tasks that are related to searched terms.

2.1.8 Site Adaptation Requirements

There are no adaptations for the web application.

2.2 Product Functions

The web application has many different functions. The first function is the task viewer and creator. The second function is the calendar module that converts the task viewer into a calendar format. The final function is that search feature that allows the user to search through the user's task and pull any data that is relevant to that searched term.

2.3 User Characteristics

The web app should be very similar to other applications on the internet. We hope to make the system as self explanatory as possible. There is not a specific age requirement but we do have an idea as to who will be use this application. This app is targeted at anybody living a busy life with a variety of tasks to perform on a regular basis, particularly students, business professionals, entrepreneurs, and freelancers.

2.4 Constraints

Any computing machines (mobile devices included) with an internet connection and an access to a modern web browser is able to view our website without any trouble. To reach as many different devices as possible and optimize the experience, our code will also account for screen scaling so that a mobile user can view our content comfortably as they would on a traditional monitor thanks to Meteor's configurable styling for reactive web pages that allows for easy scaling for a wide variety of devices. A high-speed internet connection is recommended for accessing our website for the best experience, but it can run on slower connections as well with minimal detraction from the experience.

2.5 Assumptions and Dependencies

The web application will be dependent on the javascript framework Meteor. As long as meteor does not radically change we should be able to support the web app indefinitely. Also, if we can create a secure and stable database now, we should have no problems maintaining it well into the future.

The final dependencies are on the web browsers. As browsers update and move on to new formats, we may have to alter how some of our

application is written to support the new browsers as well as any web standards that will inevitably change.

3. Specific Requirements

3.2 System Features

3.2.1 Phase I (First Release)

3.2.1.1 Landing Page

3.2.1.1.1 Login area with integration for support to login with common accounts such as Facebook, Google, Twitter, and Github as supported by Meteor's login module.

3.2.1.1.2 Successful login sends user to entries page.

3.2.1.1.3 Unsuccessful login throws error, successful login directs right to the user home page.

3.2.1.1.4 Register button for new users who don't want to login via 3rd party service.

3.2.1.1.4.1 User registers for new account with email in a valid format and valid username, then is routed to fresh entries page.

3.2.1.2 User Page

3.2.1.2.1 Basic User accounts will be created and modeled in the database to test account functionality.

3.2.1.2.2 For current phase, users can also change password or logout through same module as login

3.2.1.3 Entries Page

3.2.1.3.1 Basic "note" entry types will be created and modeled in the database to test general functionality for the first phase.

3.2.1.3.1.1 These simple entries for first phase consist of only a title to display to verify proper maintenance of user entries in the database while testing.

3.2.1.3.2 Each entry can be selected with a checkbox to mark it as complete.

3.2.1.3.3 Hide Completed Entries button at top of page

hides all entries checked as completed.

3.2.1.3.4 Delete button to remove each entry

3.2.2 Phase II

3.2.2.1 Basic Journal Entry Types

3.2.2.3.1 Support for basic journal entries will be supported, including notes, tasks, and events (all as children of the abstract “journal entry” type).

3.2.2.3.2 Notes function much like you would expect in any other simpler note-taking app with a title and a content text box, as well other special metadata (location, time, relevance to other journal entries, applied tags, etc. some of which to be implemented in later phases) that will be used for filtering and organizational purposes.

3.2.2.3.2.1 Notes Entry Boxes

3.2.2.3.2.1.1 Title

3.2.2.3.2.1.2 Description - content of note (supporting text only for now)

3.2.2.3.3 Tasks include basic fields of entry including title, description, difficulty (in terms of relative time-consumption like a jira ticket), tags to organize and relate the task to other journal entries, and the automated metadata associated with geolocation and time.

3.2.2.3.2.2 Tasks Entry Boxes

3.2.2.3.2.2.1 Title

3.2.2.3.2.2.2 Description

3.2.2.3.2.2.3 Due Date

3.2.2.3.4 Events in this phase will serve as reminders in a schedule, without any specific action(s) associated with them (although could be related to a task via a common tag).

3.2.2.3.4.1 Event Entry Boxes

3.2.2.3.4.1.1 Title

3.2.2.3.4.1.2 Description

3.2.2.3.4.1.3 Time of event

3.2.2.3.4.1.4 Location of event

3.2.2.3.5 All entries will also be tagged with appropriate metadata including the time and location that the entry was created (given appropriate user permissions).

3.2.2.2 New Entry Form

3.2.2.4.1 Triggering the new entry form leads to a basic menu for confirming an entry type (i.e. event, note, task, etc.).

3.2.2.4.2 Choosing an entry type expands the form to include all of the relevant data entry boxes for the selected entry type.

3.2.2.4.3 Selecting either ‘Cancel’ or ‘Save Entry’ at the bottom of the form will redirect back to the entries page with the appropriate changes made to the entry list.

3.2.2.4.4 Same form triggered with edit entry button if a user wants to change the values of an entry after submitting it

3.2.2.3 Search box

3.2.2.3.1 Adds ability for user to search all entries (or entries under a specific filter) to find any related to the text input given in the search bar.

3.2.2.4 Improved User Management UI/UX

3.2.2.4.1 Expand upon basic Meteor account management and improve styling of user and entry management page

3.2.3 Phase III (Future Additions)

3.2.3.1 More Account Options for User

3.2.3.1.1 Users will gain more customization over their account, particularly visually in how their entries are displayed on screen.

3.2.3.2 Expanded Entries Page Functionality

3.2.3.2.1 Expanded functionality will be added to allow for more advanced relationships between entries.

3.2.3.2.2 Creating/Editing entries now gives option to create relationship with other entries (i.e. parent-child relationships such as where a task

could include multiple subtasks such as children tasks).

3.2.3.2.3 With more complex relationships allowed between entries, more default entry types will be created

3.2.3.2.3.1 Project Entries - essentially one big task that can only be completed with the completion of certain subtasks and/or sub-events

3.2.3.2.3.2 Meeting Entries - similar to events but with expanded functionality to allow for sharing with other users on this platform as well as others through integration of third-party apps.

3.2.3.3 User Calendar Page

3.2.3.3.1 Users will be able to see any subset of the filtered entries shown on their main user page, not only in list form, but also on a calendar to better visualize entries.

3.2.3.3.2 Support will be implemented to save multiple filter presets that can be selected to quickly apply customizable filters previously set up by the user so that specific subsets of entries can be easily viewed.

3.2.3.4 UI/UX Improvements

3.2.3.1.1 After user testing, potential redesign of the website navigation menus to facilitate the ideal workflow, particularly focusing on reactivity so that the app performs optimally on small screens as well as larger ones.

3.2.3.5 Default Journal Entry Templates

3.2.3.2.1 To ease use and quicken the process of customizing a user's journal, templates will be added to help new users get the most out of the app as quickly as possible.

3.2.3.6 Integration With 3rd Party Services

3.2.3.3.1 Facebook Events integration so that users can

sync events right from their linked Facebook account without having to re-enter them.

3.2.3.3.2 Google Calendar and Office 365 integration for users to sync events and task due date reminders with their linked Google/Microsoft account(s).