Software Requirements Specification

for

OnTraq

Version 1.0 approved

Basement Brogrammers

10/22/2018

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to detail the purpose and features of the OnTraq project management tool. It will present the reader with the operating procedures and user interactions pertinent to the application’s system. This document is intended for stakeholders, users, and the other developers.

## Document Conventions

There are no specific typographical conventions that are necessary to fully comprehend this document. The bold title at the beginning of each line gives you an idea of the section that is coming up. For the diagrams, the text before it explains the specifics of it.

## Intended Audience and Reading Suggestions

This document is intended for any person, user or developer, seeking to gain a comprehensive understanding of the OnTraq system. It is recommended that the reader read this document in the order it is presented.

## Product Scope

Simply, as stated above, OnTraq is intended to be a lightweight and accessible project management tool. It will provide tools to help organize and manage development teams and the projects and individual tasks they will be working on.

OnTraq allows team members to view the tasks assigned to them in order of priority while also giving managers and overseers the ability to change the structure of the teams and project while also being able to quickly assess the state of the project.

## References

# Overall Description

## Product Perspective

OnTraq is a new, self-contained, product that does not require any other systems or software to effectively function. It allows for three types of users, the manager the member and the overseer. The main focus is to complete a very user-friendly UI that allows for easy to use analysis of the projects and its tasks.

## Product Functions

The primary functions of the system include:

* Task assignment
* Task subdivision
* Task prioritization
* Team/group assignment
* Task/project progress update
* Upload a file
* View task and project status
* Messaging between

## User Classes and Characteristics

1. **Developer**
   1. Most common user, and close second in importance
   2. Can edit task status (Completed, Started, Not Started)
   3. Can upload files of completed tasks
2. **Manager**
   1. Most important
   2. Can assign members to teams
   3. Can assign teams to projects
   4. Can divide projects into tasks and assign team members to them
   5. Can divide tasks into sub-tasks and assign team members to them
   6. Can prioritize tasks
3. **Overseer**
   1. Meant for other team members that only need to view the status of the project
   2. Can view the different dashboards and team members, but not edit them.

## Operating Environment

The environment is rather basic. The application will likely only be able to run on windows. OnTraq will not need any specific software to run.

## Design and Implementation Constraints (Optional)

The only constraint would be internet access. Most, if not all, the information will be stored on a server.

## User Documentation

No tutorial or manual is needed. The software is self-explanatory and created in order for easy use from the start.

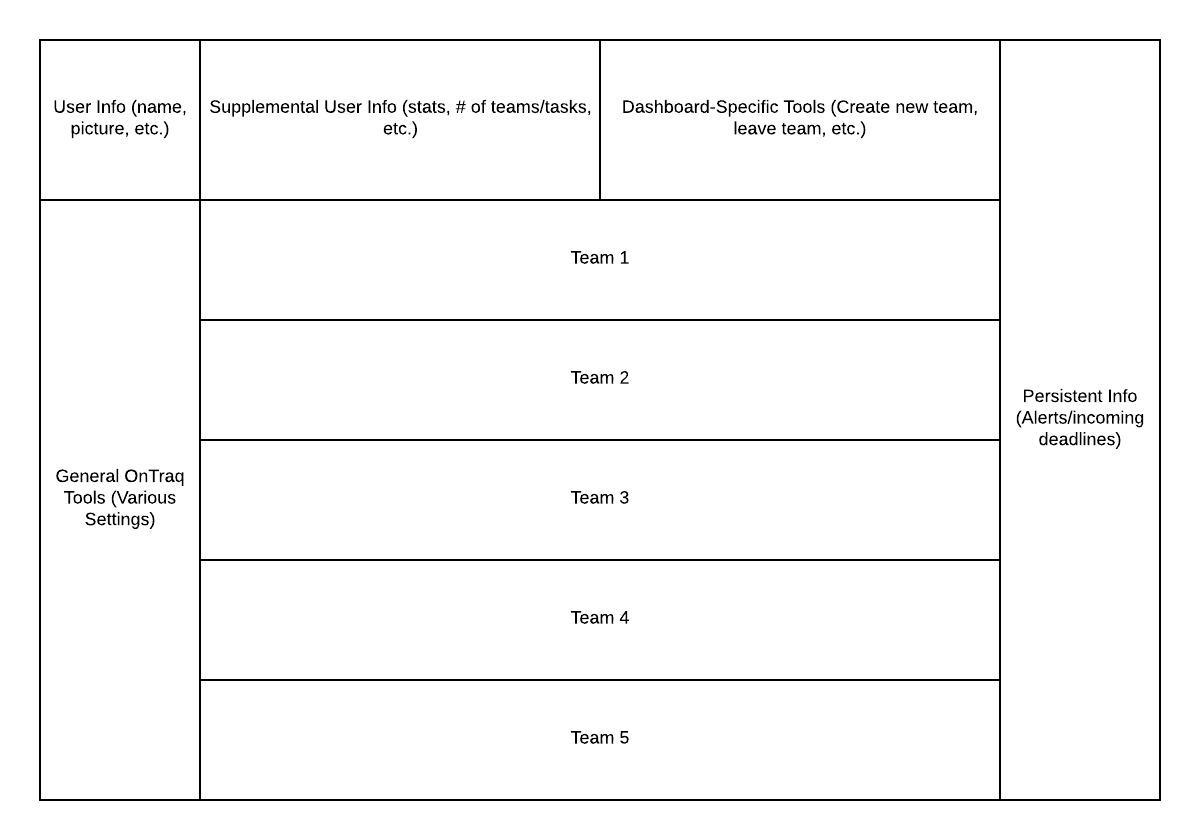
## Assumptions and Dependencies

We assume that the user will be using a machine that has been updated with the proper updates pertaining to their computer

# External Interface Requirements

## User Interfaces

The primary interface will be a series of dashboards which are nested within one another. For example, the first dashboard a user will be presented with is one consisting of a list of teams the user belongs to, along with critical information displayed along with each one. The user can select one and will be brought to another dashboard which lists all the team members within that team.



## Hardware Interfaces (Optional)

OnTraq will assume that you are running on a windows computer but, there is a possibility that it may be run on a tablet or a different android device.

## Software Interfaces

OnTraq will have an accompanying database which will store all objects (team members, tasks, etc.) necessary for functionality. The application will send changes to the server when a dashboard is closed and changes were made. The changes you make to your dashboard to tasks and files will update the team dashboard to show the rest of your status on the tasks. Each dashboard will be personally created based on your login to the system. For

## Communications Interfaces

As stated above, OnTraq will send updated versions of edited objects and overwrite the ones in the database. The formatting of these updates will be rather simple, with the server doing most of the computation and the message simply containing what object needs to be changed, and which of its parameters need to be changed.

# System Features

This section outlines the various use cases for each possible actor (Developer, Manager, Overseer) in each page of the application.

## Primary User Dashboard

**4.1.1** Description and Priority

This is the first dashboard a user will find themselves in once the program is run. It is therefore of greatest importance which shows the status of your tasks. It also shows you team names and allows you to then view the

**4.1.2** Stimulus/Response Sequences

1. Create new team
   1. User selects the option to create a new team
   2. User names the team
   3. User can add team members to the team through previous teammates or through invite code.
2. Leave team
   1. User can leave a team by selecting the option
   2. Then checking a box next to the team(s) he or she wants to leave
3. View team dashboard
   1. User can enter a new dashboard view by clicking on one of the teams he or she is a part of.
   2. View the tasks of the individual
   3. View the tasks of the team

**4.1.3** Functional Requirements

|  |  |
| --- | --- |
| **Use Case Name** | View Task |
| **Trigger** | The user has pressed to view the task and can see the description and its completion status |
| **Precondition** | The user has accessed the task and its contents |
| **Basic Path** | 1. The user selects view task 2. Can view its contents |
| **Alternate Paths** | Go back to the previous page |
| **Postcondition** | The user has successfully viewed the task |
| **Exception Paths** |  |
| **Other** |  |

|  |  |
| --- | --- |
| **Use Case Name** | Add member |
| **Trigger** | The user can add a member to the team and then set assign them a task |
| **Precondition** | The user has the capability to add a member by their OnTraq ID and give them a task. |
| **Basic Path** | 1. The use selects add member 2. Inputs member ID 3. Add the member |
| **Alternate Paths** | Go back to the previous page or cancel the addition |
| **Postcondition** | The user has successfully added a member |
| **Exception Paths** | None |
| **Other** |  |

|  |  |
| --- | --- |
| **Use Case Name** | Create Account |
| **Trigger** | The user does not have a login so they will have to create a new account |
| **Precondition** | The user doesn’t have an account |
| **Basic Path** | 1. The user selects create account 2. Imputes the information 3. Checks the password and username are valid 4. And gets added into the OnTraq database |
| **Alternate Paths** | Go back to the login page by selecting cancel. If the username they select is invalid they will get an error message. |
| **Postcondition** | The user has successfully created an account and can now login successfully in the system. |
| **Exception Paths** | Cancel the creation of the account |
| **Other** |  |

|  |  |
| --- | --- |
| **Use Case Name** | Upload file |
| **Trigger** | The user has pressed to view the task and the upload file capability |
| **Precondition** | The user has accessed the task and can now upload a file from its browser |
| **Basic Path** | 1. The user selects view task 2. Can view its contents 3. Can select to post a file to the task and mark it as completed |
| **Alternate Paths** | Go back to the previous page and view task |
| **Postcondition** | The user has successfully viewed and may or may not upload a file |
| **Exception Paths** |  |
| **Other** |  |

|  |  |
| --- | --- |
| **Use Case Name** | Login |
| **Trigger** | The user has just begun the software and now has the option to log in |
| **Precondition** | The user has already created an account within the OnTraq database and now can access their account |
| **Basic Path** | 1. The user selects to login and inputs their OnTraq ID and password 2. Can view its member dashboard |
| **Alternate Paths** | Cancel the system |
| **Postcondition** | The user has successfully logged in and now may view their |
| **Exception Paths** | None |
| **Other** |  |

|  |  |
| --- | --- |
| **Use Case Name** | View Task |
| **Trigger** | The user has pressed to view the task and can see the description and its completion status |
| **Precondition** | The user has accessed the task and its contents |
| **Basic Path** | 1. The user selects view task 2. Can view its contents |
| **Alternate Paths** | Go back to the previous page |
| **Postcondition** | The user has successfully viewed the task |
| **Exception Paths** |  |
| **Other** |  |

|  |  |
| --- | --- |
| **Use Case Name** | Progress bar |
| **Trigger** | Each task as you upload files that are complete the status bar increase |
| **Precondition** | The task needs to get completed |
| **Basic Path** | 1. Selects the task and can view the task completion status 2. Can view its contents and see the progress bar |
| **Alternate Paths** | Complete the task |
| **Postcondition** | The user has successfully viewed the task |
| **Exception Paths** | None |
| **Other** |  |

\

|  |  |
| --- | --- |
| **Use Case Name** | Full team updates on tasks |
| **Trigger** | When the user uploads a file it can be viewed by everyone on the team. |
| **Precondition** | The user has to have a task and ready to submit his current work |
| **Basic Path** | 1. The user selects view task 2. Can view its contents 3. Uploads a file 4. Updates the database to all the teams dashboard |
| **Alternate Paths** | Go back to the previous page or the member dashboard |
| **Postcondition** | The user has successfully viewed the task and uploaded the file to all the members of the team |
| **Exception Paths** | None |
| **Other** |  |

## Team

|  |  |
| --- | --- |
| **Use Case Name** | View Team |
| **Trigger** | The user has pressed view team but and can see the team that he is apart of |
| **Precondition** | The user has signed in and can now see their team and the list of members and can switch from task to team view |
| **Basic Path** | 1. The user selects the team dashboard 2. Can view the team members 3. Swap to task view and view the tasks |
| **Alternate Paths** | Go back to the previous page |
| **Postcondition** | The user has successfully viewed the team members |
| **Exception Paths** | Can view task view and member view |
| **Other** |  |

## Dashboard

**4.2.1** Description

This is the dashboard detailing all of the members of the team. The user can also change between member view and task view, displaying all of the tasks assigned to the team.

**4.2.2** Stimulus/Response Sequences

* 1. Add/remove members
     1. Allows the user to add a new member to the group if they have been granted managerial permissions
  2. Switch to task view
     1. Can change to task view which displays all of the tasks which have been assigned to the group.
  3. View member
     1. Can change to the dashboard of an individual member.
     2. Can view the status on the users individual task
     3. Can view the entirety of the users task

**4.3 Member Dashboard**

4.3.1

This is the dashboard where the user can view the information specific to the selected member including member statistics, the tasks they are working on, and any groups they belong to.

4.3.2

* 1. View subtasks
     1. View the smaller components of the task that it has been divided into.
  2. Set task priority
     1. Set how important the task is in relevance to other tasks.
  3. Upload version (if applicable)
     1. If the task is something that can be completed and uploaded, the user can upload the latest version here.
  4. Set task status
     1. The user can change the task status between not started, started, and completed.

# Other Nonfunctional Requirements

## Performance Requirements

The system needs to be able to handle multiple different users running the software at once from different computers and different machines. It has to be able to deal with high traffic times where a lot of users are doing work on their tasks. For example, if multiple different teams are working on completing tasks or logging on the software at the same time and we want to be able to handle the heavy traffic on the softare.

## Security Requirements

In the final version of the product, we will have encrypted communication between the server and the client. Additionally, the program will require a secure login. And will be stored on the local database.

## Software Quality Attributes

The program won’t demand any specific attributes, the interface also won’t allow for much confusion. We will be able to run updates due to the way we have constructed our program and its variability.

# Other Requirements

OnTraq will have multiple databases that are going to store the users name, Identification number,

Appendix A: Glossary

|  |  |
| --- | --- |
| Term | Definition |
| Task | This is the individual assignment that is assigned to a member |
| Subtask | This separates a task and delegates parts of the task to other members |
| Task View | This is a view shows you all of the tasks for either the member or the entire team. |
| Dashboard | The view of the user of the system. |
| Developer | The most basic user that allows to work on tasks and view teams and status of their own tasks, also can create a team, leave and add team. |
| Manager | This user can assign tasks; modify teams; create teams; delegate tasks |
| Overseer | This user views the status of all the teams and mangers that he has control over. |
| OnTraq ID | The identification that allows the Ontraq users to be identified by a number, rather then a username because people could have the same name. |

Appendix B: Analysis Models

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

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>