Software Requirements Specification

for

Tutor Application

Version 1.0 approved

Prepared by <author>

<organization>

<date created>

Table of Contents

Table of Contents 2

Revision History 2

1. Introduction 1
   1. Purpose 1
   2. Document Conventions 1
   3. Intended Audience and Reading Suggestions 1
   4. Product Scope 1
   5. References 1
2. Overall Description – Chase Pisone 2
   1. Product Perspective 2
   2. Product Functions 2
   3. User Classes and Characteristics 2
   4. Operating Environment 2
   5. Design and Implementation Constraints 3
   6. User Documentation 3
   7. Assumptions and Dependencies 3
3. External Interface Requirements 3
   1. User Interfaces 3
   2. Hardware Interfaces 4
   3. Software Interfaces 4
   4. Communications Interfaces 4
4. System Features - Klei Bendo 4
   1. Private Messaging 4
   2. Direct Deposit 6
   3. Find a Tutor 7
   4. Create Profile 8

4.5. Apply for Tutoring 10

1. Other Nonfunctional Requirements 11
   1. Performance Requirements 11
   2. Safety Requirements 11
   3. Security Requirements 11
   4. Software Quality Attributes 11
   5. Business Rules 12
2. Other Requirements 12

Appendix A: Glossary 12

Appendix B: Analysis Models 12

Appendix C: To Be Determined List 12

Revision History

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

# Introduction

## Purpose

This software is an application to connect tutors and tutees. This SRS covers the entire scope of the software and all parts of the system. The software requirements for this application are specified in this document. The revision and release number are to be determined.

## Document Conventions

No standard conventions have been used throughout this document. Every requirement statement in this document has its own priority unless otherwise stated.

## Intended Audience and Reading Suggestions

This document may have many different types of readers. These include application managers, developers, tutor users, and tutee users. A general user (tutor or tutee) should read this document in the following order: Overall Description, System Features, Other Nonfunctional Requirements. An application manager should read all sections of this document in the order of the Table of Contents, found on page 2.

## Product Scope

This software is an application to connect tutors and tutees. Tutors and tutees are conveniently matched to increase the academic and professional success of all parties. The goal of this application is to best match tutees with appropriate tutors and facilitate the tutoring relationship between parties.

## References

References will be updated and cataloged here.

# Overall Description

## Product Perspective

The product is supposed to be a tool to for users to find and acquire help/tutoring on resources related to college courses. This is a web based system aimed at easily connecting students to tutors with minimal complexity.

## Product Functions

Major functions of the product are:

* User accounts: The system allows the creation of accounts by users, and also provide features to view and update profile accounts.
* Search: allows the user to search through courses and tutors to find one that best fits the user.
* Chat: Chat is for the user and the tutor to be able to communicate.

<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

## User Classes and Characteristics

There are three main user classes for this application: Tutor, Tutee, and application manager. Tutor users frequently use the application and heavily rely on product functions to respond to listings and chat with tutees. Tutee users may use the application heavily to communicate with tutors and post listings each semester. Application managers regularly check the application and provide updates. The application managers are responsible to ensuring that the application meets all requirements.

## Operating Environment

This software will operate as a web-based program. The application will be compatible with all major operating systems and web browsing platforms, including: Safari, Firefox, and Chrome.

## Design and Implementation Constraints

The development of this application will take place with 2 week sprint cycles, each cycle ending with a deliverable. The development team is responsible for maintain the delivered software, and ensuring that the software meets all programing standards, conventions, and security concerns. Memory requirements regarding this application have yet to be determined.

## User Documentation

Upon first (downloading or opening/creating an account) the user will be given a simple tutorial on the main features of the product. This tutorial will be accessible by the user at any time they wish to return to it. The application will also have a Q&A section to answer common questions for the user.

## Assumptions and Dependencies

This software depends on the user having knowledge of web-based applications, and access to the web. The software does not currently have any dependencies on external factors.

# External Interface Requirements

## User Interfaces

When the application is launched, a log in screen containing user name and password text fields, a sign in button, and the name off the application will appear. After signing in the user will be taken to the home page. The main way to navigate around is a menu that takes a portion of the left side of the screen. This menu will have options to direct you to the various features of the application such as posting listings, searching for tutors, viewing your profile, etc. This menu will vary in features depending on if the user is a student or tutor, and will be visible no matter where you are in the application.

## Hardware Interfaces

Local data for the interface will be stored in variables on the users system while sensitive back end data will be stored on a server.

## Software Interfaces

Email will be needed to send updates and information to the users such as scheduled meeting times. Data such as user profiles, schedules, and what type the user is will be store in the back end.

## Communications Interfaces

Communications used by the application will need an interface to connect to the backend which will use a JavaScript add on. HTTP will be used since the application is a website.

# System Features

## Private Messaging

4.1.1 Description and Priority

All users should be able to privately message each other through the application so that they can connect with each other. This is a high priority feature.

* Benefit: Easier communication
* Penalty: None
* Costs: UI design, Database Management, Encryption
* Risks: Malware attacks, identity stealing, account hacks

4.1.2 Stimulus/Response Sequences

* User A opens User B’s profile and clicks the private message icon on their profile
* System opens a chat window between both users.
* User A enters their message.
* User A presses the deliver message button.
* System encrypts the plaintext to ciphertext entered and sends it to user B.
* System decrypts the ciphertext back to plaintext once user B receives the message.
* System notifies user B through a pop up icon.
* User B clicks / taps the icon and opens the message.

4.1.3 Functional Requirements

REQ-1: The system shall warn the receiver of a message if they want to engage with potentially malicious content, in the event of trying to access an external source that the message contains (for example: a link, photo, file).

REQ-2: The system shall ensure that only authorized users are able to view their private messages.

REQ-3: When prompted, the system shall report to the user a list of all their private messages, ranked from the most recent date that a message was sent / received.

REQ-4: The system shall notify the receiver that a new message is available to view within 1 second of receiving the message.

## Direct Deposit

4.2.1 Description and Priority

This feature enables all users to enroll into direct deposit in order to receive and send their payments. This function is low priority.

4.2.2 Stimulus/Response Sequences

* User presses the “Direct Deposit” button from their menu.
* System opens up a new window that contains a form that the user needs to fill in with their bank account information.
* User fills in their bank account information.
* User presses the ‘enroll’ button.
* System checks whether the information provided is correct.
* If information provided is correct:
  + System saves the user’s bank account information
* If information provided is incorrect:
  + System asks the user to insert their bank details again.

4.2.3 Functional Requirements

REQ-1: The system shall ensure that only the authorized user is able to access their bank details

REQ-2: The system shall notify the user if any banking information is missing after they try to enroll into direct deposit.

REQ-3: The system shall notify the user if their banking information entered is incorrect after they try to enroll into direct deposit.

REQ-4: The system shall keep an activity log of every user’s transactions within the app, updated every 3 hours

REQ-5: The system shall ensure that the user has access to their activity log of transactions.

## Find a Tutor

4.3.1 Description and Priority

This feature will enable all users to search for a tutor by name or subject. This feature is of high priority.

4.3.2 Stimulus/Response Sequences

Sequence 1:

* User presses the ‘Find Tutor’ button in their menu.
* System shows the user two options to pick from: ‘Search by name’ , ‘Search by subject’.
* User clicks / taps ‘Search by name’ button.
* System opens up a new search box.
* User enters their text in the search box.
* User clicks / taps the search button.
* System checks whether the name matches to any tutor profile.
* System shows all the available tutor profiles (if any) that match that name.
* User presses the ‘Find Tutor’ button in their menu.
* System shows the user two options to pick from: ‘Search by name’ , ‘Search by subject’.
* User clicks / taps ‘Search by subject button’.
* System checks whether there is any subject with that name registered.
* System shows all the available tutor profiles (if any) for that subject.

4.3.3 Functional Requirements

REQ-1: The system shall notify the user if their search by subject resulted in no findings, within 3 seconds of their search initiation.

REQ-2: The system shall notify the user if their search by name resulted in no findings, within 3 seconds of their search initiation.

## Create Profile

4.4.1 Description and Priority

This feature will enable all users who are logged in to create their own profiles. Based on what they want to do with the app they will be able to create a student or a tutor profile. This feature is high priority.

4.4.2 Stimulus/Response Sequences

Sequence 1

* User clicks / taps the ‘Create Profile’ button on their menu.
* System shows to the user two options: ‘Create student profile’, ‘Create tutor profile’
* User clicks / taps the ‘Create student profile’ button
* System opens a new window and asks the user to insert their full name, student ID and email
* User enters their full name, student ID and email
* User clicks/ taps the ‘Submit’ button
* System checks whether the user is a student at the University of Maine based on the information submitted
* If the user is a student at UMaine, then the system opens up a new window containing a form that the user needs to fill with their student profile details.
  + User fills in the rest of their profile
  + User clicks / taps the ‘Create profile’ button
  + System saves user’s information.
* If the user is not verified to be a UMaine student, the system will ask the user to insert their student information again (up to 3 tries), until they insert the correct information.

Sequence 2

* User clicks / taps the ‘Create Profile’ button on their menu.
* System shows to the user two options: ‘Create student profile’, ‘Create tutor profile’
* User clicks / taps the ‘Create tutor profile’ button
* System checks whether the user has applied to become a tutor on this app.
* If the user has applied:
  + System checks if their application is approved, declined, or under review.
  + If their application has been approved:
    - System opens up a new window containing a form that the user needs to fill in with their tutor profile details.
    - User fills in the rest of their profile
    - User clicks / taps the ‘Create profile’ button
    - System saves user’s information.
  + If their application has been declined:
    - System notifies the user that they can’t create a tutor profile on this application.
  + If their application is under review:
    - System notifies the user to wait for the result of their application.
* If the user has not applied:
  + System notifies the user to apply for a tutor first before creating a tutor profile.

4.4.3 Functional Requirements

REQ-1: The system shall notify a user that has not filled all required information in the form to do so after trying to create their profile.

REQ-2: The system shall ensure that only UMaine students can create a student profile.

REQ-3: The system shall ensure that only users that have been approved of tutoring are able to create a tutor profile.

## 4.5. Apply for Tutoring

4.5.1 Description and Priority

This feature enables all users who want to become a tutor on this app to apply for this position. This is a high priority feature.

4.5.2 Stimulus/Response Sequences

* + User clicks ‘Apply to Tutor’ button on the home page
  + System prompts user for various questions and records the answer
  + Application managers review applicant answers
  + Application manager updates the system is the user’s application has been accepted or declined.

4.5.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1: The system shall notify the user if they do not answer a required question.

REQ-2: The system shall send the user an email once their application status has been updated.

REQ-3: The system shall alert the application manager weekly with applications that need to be reviewed.

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

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.>