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.

<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SRS, particularly if this SRS describes only part of the system or a single subsystem.>

## Document Conventions

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

## Intended Audience and Reading Suggestions

This document may have many different types of readers. These include application managers, developers, tutor users, and tutee users.

<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.>

## 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.

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

## References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.>

# Overall Description – Chase Pisone

## 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 or android based whatever we pick) system aiming to easily connect users to tutors with minimal complexity.

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

## 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: Search is for 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

The user is expected to be able to navigate the internet (or work an app) in order to access the service. Account administrators are expected to be familiar with the features of the (app/website).

<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.>

## Operating Environment

To be determined. Either Android or Website

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

## Design and Implementation Constraints

Time limitations are the deliverables or course schedule for how far we need to be developed.

\*Must be a University of Maine student to use the product.

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>

## 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.

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

## Assumptions and Dependencies

This software depends on either Android or a version of browser installed on the system (depends on which one we do)

<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).>

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features - Klei Bendo

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## Private Messaging

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

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

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

* 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

<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 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

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.2.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

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

4.2.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

* 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

<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 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

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.3.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

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

4.3.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

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

<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 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

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.4.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

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.

4.4.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

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 right now.
  + 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

<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 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

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.5.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

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

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

TBD

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.>

TBD

# 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.>