Requirements Specification

Cat Scheduler

FIDO

Table of Contents

1. Introduction

This section gives a scope description and overview of everything included in this SRS document. The purpose for this document is described in this section.

1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the “Fully Integrated Dog Organizer” (FIDO) software. It will give the purpose and complete declaration for the development of the system. Also it will detail the system constraints, interface, and interactions with other external applications.

1.2 Document conventions

FIDO

“Fully Integrated Dog Scheduler”

CAT

“Computer Action Team”

1.3 Intended audience

This document is primarily intended to be proposed to the sponsor for its approval, to the instructor Bart for verification of the document, and as a reference for developing the first version of the system for the development team. Also may be used to allow testers to determine if the features are working correctly.

1.4 Additional information

The tool will be call “Fido”. Standing for “Fully Integrated Dog Scheduler”.

1.5 Contact information/SRS team members

Name	**Slack User**	**Team Role**	**Phone Number**	**Email**
Alexander Simchuk	bowzr	Communications	(503)-928-1829	sim6@pdx.edu
Cody Wyatt	cody	Specifications	(503)-816-9364	cwyatt@pdx.edu
Eiyad Alkadi	ealkadi	Taskmaster	(713)-829-8502	ealkadi@pdx.edu
Graham Drakeley	drakeley	Project Architect	(503)-367-2337	gt.drakeley@gmail.com

Isaac	maine	Infrastructure	(503)-707-2692	iarcher@pdx.edu
Jonny Castle	castlez	Team Lead	(503)-913-6155	castle2@pdx.edu
Nima S	nima	Risk Management	(503)-858-2752	seyed@gmail.com

1.6 References

cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf
Used as assistance in creating this document and ensure accuracy.

2. Overall Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

2.1 Product perspective

This system will consist of several parts: two interfaces and one database. There will be one interface to view availability and information of students along with allowing for creation and retrieval of term blocks. Another interface will be for the client interface which will allow for updating availability, viewing availability for a specific term, and retrieving a specific term. The client then should be able to select a term, have a week view for submission, a shift preference, notifications section, and be able to submit their changes. The database should have several parts, one part to hold the terms, one for the students, and one for the availability submitted by the students. The interfaces will have to communicate to the database to create, retrieve, or modify the data.

2.2 Product functions

Fido is meant to be a remade implementation of the current CAT scheduler for which availability is recorded for the members of the CAT by the user submitting their own availability. For this an administrative interface will be made to allow for creating terms for which the clients or students will be able to set their availability, along with setting due dates for the submissions for the specified terms. Clients should be able to submit and update/modify their availability up until that due date. After the clients submit their availability the administrator should be able to see the submissions of that client and the other clients that have also submitted that information, also allowing to see the clients that have not submitted anything and still need to before the due date.

2.3 User classes and characteristics

There will be two types of users that interact with the system: students/clients to submit availability and administrators to view that info and create term blocks. Each of these two types of users has different use of the system so each of them has their own requirements.

The administrators will not be modifying the database directly but rather using the admin interface to view student data along with modifying terms.

The clients will not have direct access to the database but will be able to modify their own availability with the client interface to allow for submitting availability without having the need to be in person.

3. External Interface Requirements

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface

3.1 User interfaces

A user of the interface should first be asked to log in if not already. They should then be taken to the page where they are able to see what terms are available to set availability for and also be able to select a term and then either view or enter their availability.

An administrator of the interface should also be asked to log in if not already and then be taken to the admin interface. There they should be able to Add a term for availability to be set along with a due date for that term. They should be able to view the terms of availability and should be able to select the term by name. For adding a term the admin should be able to Set the name for the term, for example the specified term followed by year, Summer 2016, along with the dates that term has for the start and the ending. There should also be a due date so clients are not able to change their availability after the due date and will be finalized for that term.

3.2 Hardware interfaces

Since there will be web interfaces there will not be any direct hardware interfaces.

3.3 Software interfaces

The interfaces will interact with the database in order to read the required information or modify any information within the database.

3.4 Communication protocols and interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems.

4. System Features

4.1 Administrative Interface

4.1.1 Description and priority

The interface will allow administrators to set up terms along with viewing terms and the submitted client schedules for use to make a schedule to be used with existing software.

4.1.2 Action/result

Action: Admin wants to add a term
Result: The system stores the data within the form and adds it to the database and to the client interface for use.

Action: Admin wants to view a term
Result: After the system asks for which term to query it then shows the terms that are available to view and will display the student info for selected term.

4.1.3 Functional requirements

Database script to get student availabilities
Database script to get student information
Database script to create a term block
Database script retrieve/get term block

4.2 Client Interface

4.2.1 Description and priority

The client interface will allow clients to see what terms are available for submissions, see previous terms, to set or update availability for open terms, and to bring up previously submitted data from previous term.

4.2.2 Action/result

Action: Client want to see available terms

Result: The system checks the database and returns the terms that are able to be modified by the client and displays those terms to client.

Action: Client wants to see previous terms availability
Result: The system finds the term selected and returns the data from the database corresponding to that term.

Action: Client wants to set a new availability to selected open term
Result: The system will check to make sure valid data has been entered and then sends that data to be added to the database

Action: Client wants to modify or update availability for selected open term
Result: The system will validate the data, go into the database and find previously entered data and overwrite the data there to the newly enter data.

Action: Client wants to see what they previously entered for availability
Result: The system will go into the database and load the data for their availability from the selected term if any and load that data into the interface.

Action: Client has a preference of shift
Result: If the client is done with mentor sessions the option will be available once the client has been flagged they are done with that, then the system will set the preference to the client to the specified shift preference.

4.2.3 Functional requirements

5. Other Nonfunctional Requirements

5.1 Performance requirements

Performance for this should allow the clients and administrators to enter data and have that data be input into the database in a reasonable time for other actions to be done shortly after.

5.2 Safety requirements

The database and interfaces should not cause any safety problems.

5.3 Security requirements

Client information should be secure and only administrators should be able to access their data. Authentication will be required for both the administrators and clients to ensure the data being entered is authentic and availability is done by the clients themselves and not others.

5.4 Software quality attributes

The software should be web based and available to any web browser. Fido will be written in php to allow for maintainability as requested. The database should account for reusability to allow for many uses and will also allow for testing. Usability should exceed that of the current system and clients should be able to set their availability with little to no issues. This system is being designed to make setting availability less of a struggle and less error prone while also easy to use.

5.5 Project documentation

Documentation for this project are currently stored on Google Docs to allow for single copies of the documents to prevent from old files being used and to allow for editing documents online. This should also prevent the problems of backing up the documents to other places.

5.6 User documentation

User documentation will later be added to the finished product if needed.

6. Other Requirements

The system must and will follow the Apache Version 2 License.

Appendix A: Terminology/Glossary/Definitions list

FIDO
: “Fully Integrated Dog Scheduler”

CAT
: “Computer Action Team”

Client
: Referring to the users setting their own availability. May also be ‘Students’.