Team Blue

Career Tracker

Software Architecture Specification

William Ntumba, Scott Shrout, Anthony Freitas

Table of Contents

1.1 Overview 2
1.1.1 Identification 2
1.1.2 Goals and Objectives 2
1.1.3 Compatibility and Interoperability 2
1.1.4 Extensibility 2
1.1.5 Architectural Model 3
1.1.5.1 Architectural Concepts and Representations 3
1.1.6 Software Architectural Overview 3
1.1.6.1 Architecture Block Diagram 3
1.1.6.2 Architecture Block Diagram Descriptions 4
1.2 Subsystem Decomposition 5
1.2.1.1 Subsystem Decomposition Model 5
1.2.1.2 Subsystem Decomposition Model Descriptions 5
1.2.2.1 UML Component Diagram 6
1.2.2.2 UML Component Diagram Descriptions 6-7
1.3 Hardware/Software Mapping 8
1.3.1.1 Deployment Diagram 8
1.3.1.2 Deployment Diagram Descriptions 9
1.4 Persistent Data Management 10
1.4.1.1 Database Diagram 10
1.4.1.2 Database Table Descriptions 11
1.5 Access Control and Security 12
1.5.1.1 User Operations 12
1.5.1.2 Authentication and Security Provisions 12
1.5.1.3 User Authentication Flow Chart 12
1.6 Global Software Control 13
1.7 Boundary Conditions 14
Individual Contributions 16
Key Personnel Information 16

1. Proposed Software Architecture

1.1 Overview

1.1.1 Identification

This document describes the software architecture proposed for the Career Tracker web application to be used to support tracking of data elements related to post-graduates of the Indiana University Southeast School of Natural Sciences for reporting and analytical purposes by the Office of the Dean.

While this product is intended to address the specific needs of the aforementioned, it is possible the application could be used by other university departments in some capacity.

1.1.2 Goals and Objectives

The goal of this document is to detail all relevant software components included in this product, including their responsibilities, interfaces, and behaviors.

1.1.3 Compatibility and Interoperability

This product is built using application frameworks utilized throughout the industry to deliver similar content to end users. This web application is intended to be compatible with all modern operating systems and web browsers being utilized by the university at present and for the foreseeable future. It is intended for use on desktop and laptop personal computing devices.

In addition, the server-side applications used in storing and serving content, including Node JS and MySQL, are available open source for a variety of modern operating systems, including those commonly used in hosting web applications.

1.1.4 Extensibility

As the application's core functionality exists within editable files written in HTML, Typescript, and JavaScript and is documented in detail herein, it is conceivable that the application's components could be extended to include additional features over time.

1.1.5 Architectural Model

Functional block diagrams are used to represent the architectural model of this product.

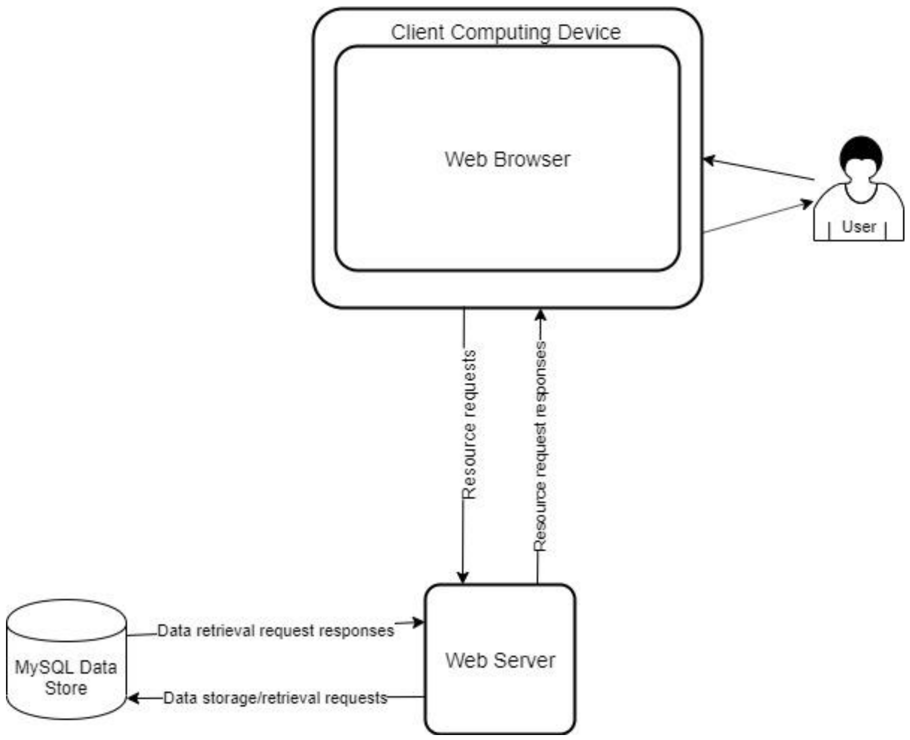
1.1.5.1 Architectural Concepts and Representations

The functional block diagrams found in this document detail the architecture of various software components.   Boxes shown in the diagrams represent distinct functional software or hardware components. Interfaces between components are represented by lines or arrows.

1.1.6 Software Architectural Overview

Contained within this section is a block diagram showing the major components of the overall software architecture. It is intended to only describe components at a high level. Detailed architectural specifications are contained within the proceeding sections of this document.

1.1.6.1 Architecture Block Diagram


Client Computing Device
Web Browser
User
Resource requests
Resource request responses
Data retrieval request responses
Web Server
MySQL Data Store
Data storage/retrieval requests

1.1.6.2 Architecture Block Diagram Descriptions

Web Server – The web server compromises the hardware and software required to serve web resource content to client applications.

> **MySQL Data Store** – The MySQL Data Store is a relational database management system (RDBMS) operating on a database server.
>
> **Web Server** – The web server listens for, and responds to, incoming web resource requests; Node JS allows for server-side applications written in Javascript.

Client Computing Device - The computing device, such as a desktop or laptop personal computer, is utilized by the user to access the web application.

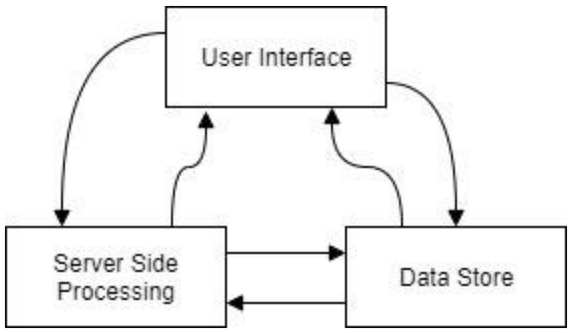
> **Web Browser** – The web browser is a Javascript-enabled web client application on the client computing device used to initiate web resource requests and render response data for user consumption.

User – The user is a university staff member utilizing the application to store and retrieve data.

1.2 Subsystem Decomposition

1.2.1.1 Subsystem Decomposition Model

The subsystem decomposition model shown below depicts the major subsystems contained within and interacting with the application. Interactions among the subsystems are indicated with arrowed lines indicating the general flow of information.



1.2.1.2 Subsystem Decomposition Model Descriptions

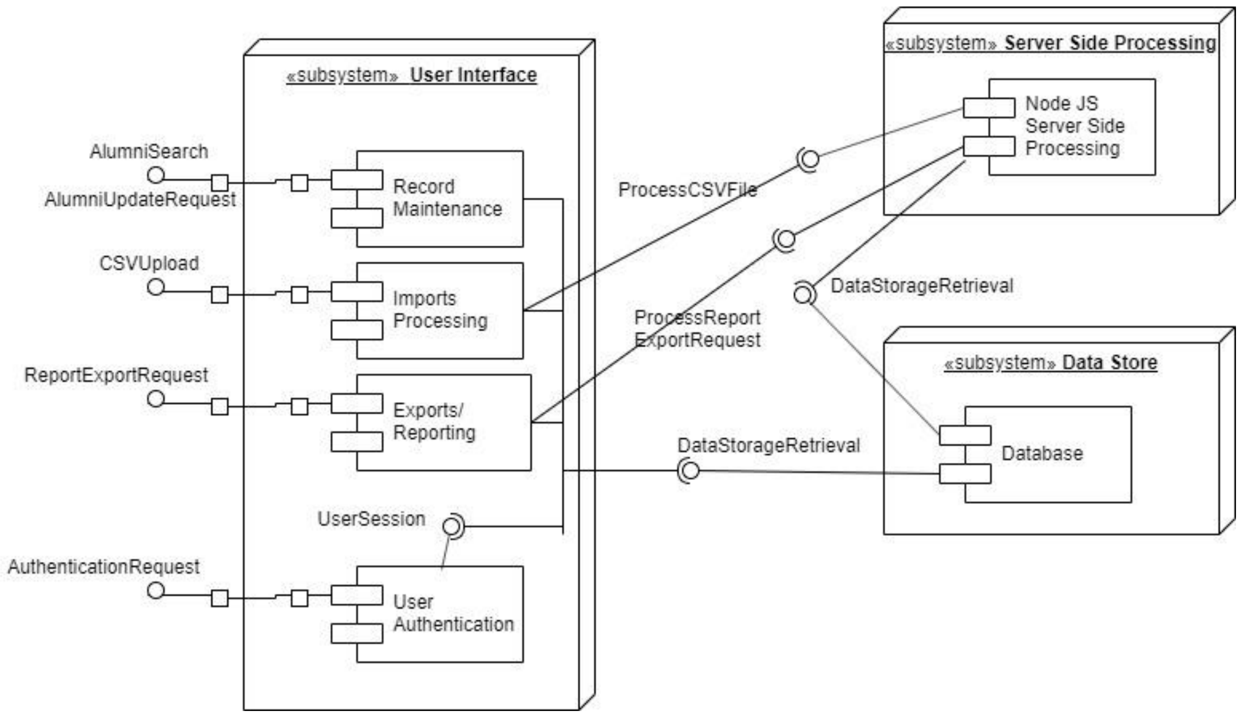
User Interface – The user interface subsystem is responsible for facilitating interaction between the end user and the application. It allows the user to authentication, view/edit alumni information, submit import requests, and submit reporting/export requests.

Server Side Processing – The server side processing subsystem processes import and export/reporting requests. For export requests, it retrieves and formats the data as needed and returns the data to the user interface for rendering or download. For import requests, it reads the submitted file and applies needed updates to the data store.

Data Store – The data store subsystem allows for storage and retrieval of persistent data used in the application.

1.2.2.1 UML Component Diagram

The component diagram below depicts the major components of the application, including their interfaces and dependencies.



1.2.2.2 UML Component Diagram Descriptions

User Interface – The user interface is responsible for facilitating interaction between the end user and the application. The user interface in this project will be accessed within a user's web browser.

Record Maintenance – The record maintenance component allows a user to search for, view, and make updates to records relevant to alumni, including alumni general/contact information, jobs/employers, post-graduate plans, and comments.
Imports Processing – The imports processing component is responsible accepting CSV file upload requests from user and passing request to server side processing subsystem for processing.

Exports/Reporting – The exports/reporting component accepts criteria from the user, submits request to server side processing subsystem for requested data, and renders data or pushes file to client for download.

User Authentication – The user authentication component accepts authentication requests from users, validates submitted credentials, and establishes a user session.

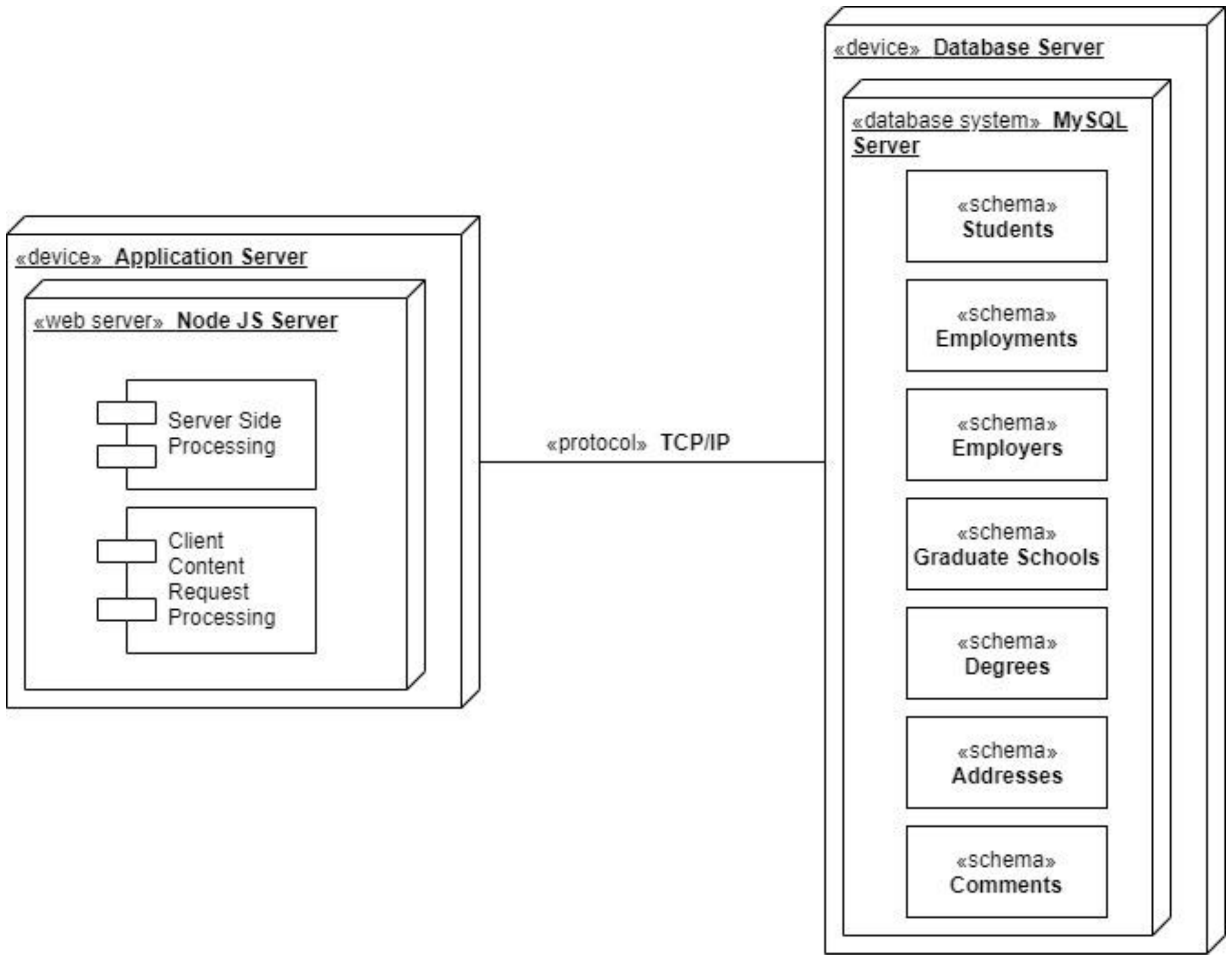
Server Side Processing – The server side processing component is responsible for processing import and export/reporting requests submitted by users.

Data Store – The data store component is responsible for storing and retrieving data needed by the application.

1.3 Hardware/Software Mapping

1.3.1.1 Deployment Diagram

The deployment diagram in this section depicts the physical configuration of the software and hardware used in the application.



1.3.1.2 Deployment Diagram Descriptions

Application Server – The application server is the hardware device on which the web server application runs.

> **Node JS Server** – Node JS Server is an application running on the application server that receives and responds to web client requests.
>
> > **Server Side Processing** – Node JS processes requests for content or file processing requests using pre-defined libraries.
> >
> > **Client Content Request Processing** – The web server listens for and responds to requests from clients for web content.

TCP/IP – Transmission control/internet protocol used to establish communication between application and database servers.

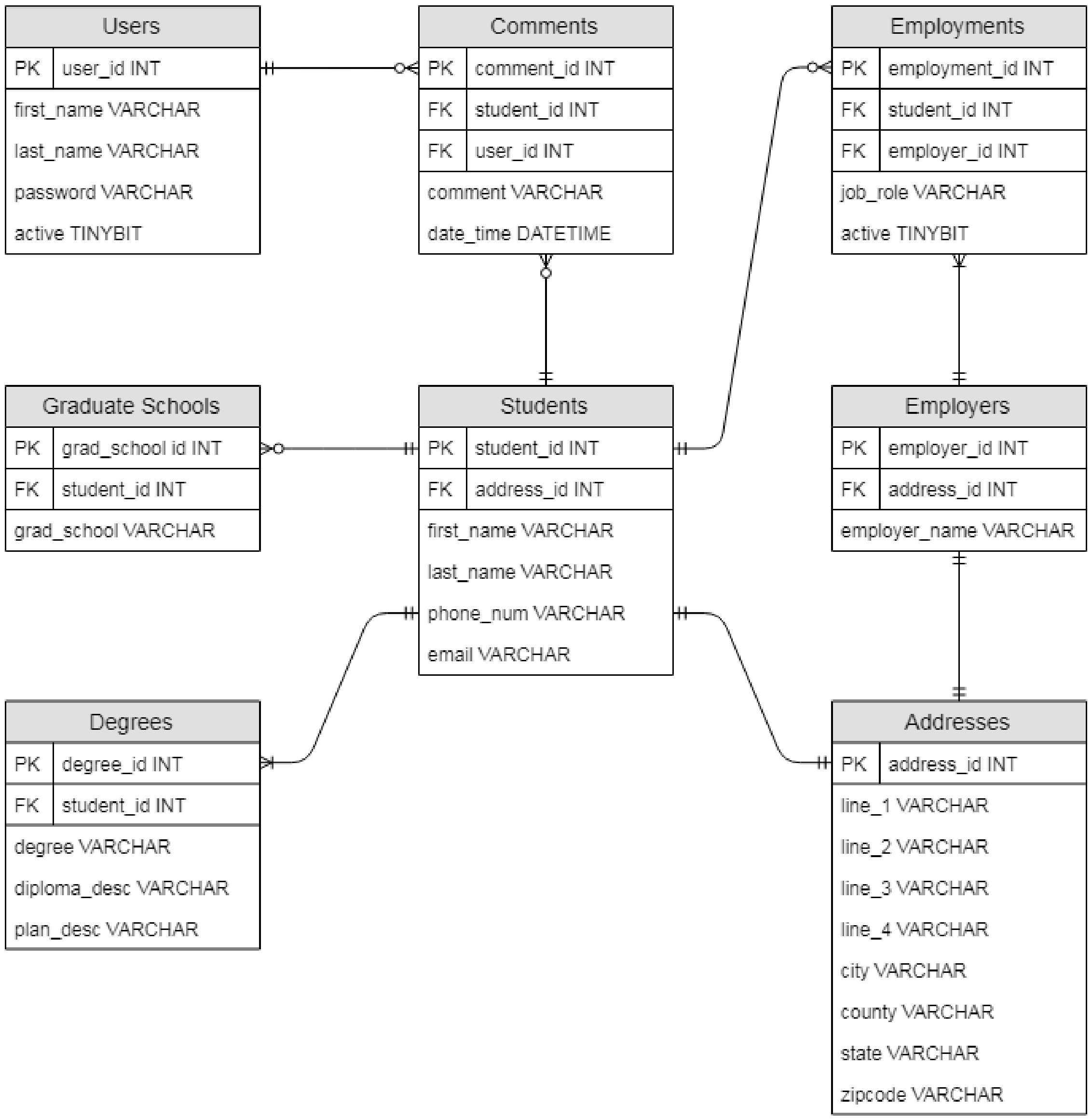
Database Server – The database server is the hardware device on which the relational database management system (RDBMS) application runs.

> **MySQL Server** – MySQL Server is a relational database management system used to store and retrieve data using the application
>
> > **Students** – The Students table contains general student information, including name, phone number, e-mail address, and address identifier. The student_id is used as a foreign key in several other related tables.
> >
> > **Employments –** The Employments table contains job entries for a student, including the role name and employer.
> >
> > **Employers –** The Employers table contains the name and address for employers referenced in the Employments table.
> >
> > **Graduate Schools** – The Graduate Schools table contains the name of graduate schools associated with a student.
> >
> > **Degrees** – The Degrees table contains degrees attained by students, including plan and diploma descriptions.
> >
> > **Addresses** – The Addresses table contains address information related to students and employers.
> >
> > **Comments** – The Comments table contains comments entered by a user related to a Student record.

1.4 Persistent Data Management

1.4.1.1 Database Diagram

The persistent data in this application will be the student and user information stored in the MySQL database. The diagram below presents a visual overview of the relationship between the entities in the database and the attributes associated with them.



1.4.1.2 Database Table Descriptions

Students – Every student record will have fields for the student's first name, last name, phone number, and email.

Degrees – A degree record is mandatory for each graduated student and students may have multiple. This category contains fields to list a degree that the student has, a description of the degree, and a plan description.

Graduate Schools – A graduate school record is optional for students and students may have multiple. There is a single field here to list a graduate school that the student has attended.

Employments – An employment record is optional for each student and students may have multiple. There is a field to list a student's place of employment and an active indicator to identify if the student is currently working there.

Employers – Each employments record should have an employer. There is a single field in this category for the employer/company's name.

Addresses – Every student and employer should have one current address record. There are four fields labeled line 1-4 for entering a street address, apartment number, and other relevant information. The other fields are for the remaining parts of the address.

Users – User records will be created when a new user adds a name and password to the system for login purposes. The fields here list the user's first and last name, their password, and an indicator to identify if that user is currently active.

Comments – This is an optional record that can be added to students for various note taking purposes. Each record will have a field for a comment and a date/time for that comment.

1.5 Access Control and Security

1.5.1.1 User Operations

Import Alumni Data – A file containing alumni data will be uploaded to the database

Export Alumni Data – The user will download a file that contains the requested alumni data

Edit Alumni Record – The user will modify an existing alumni record in the database

Delete Alumni Record – The selected alumni record in the database will be removed

View Alumni Records – Data will be displayed according to the user's search criteria

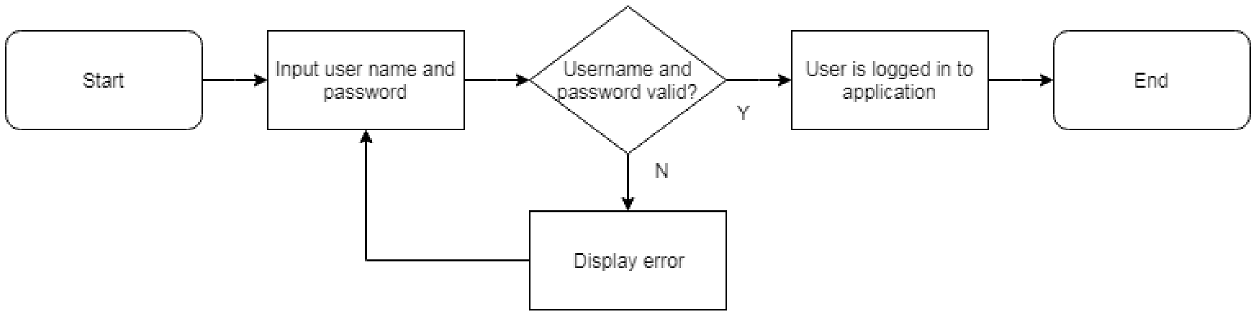
Display Alumni Report – The requested report will be displayed to the user

1.5.1.2 Authentication and Security Provisions

All potential users of the application will need to create a name and password to be able to log in to the alumni database application. Once authenticated, users will then be able to perform any of the operations above. If a user cannot be authenticated, they will remain at the log in screen and have the option to input their name and password again.

1.5.1.3 User Authentication Flow Chart

The chart below illustrates the process of user authentication.


Start
Input user name and password
Username and password valid?
Y
N
User is logged in to application
End
Display error

1.6 Global Software Control

This section describes the global software control flow, or the way actions take place within the system in a sequential manner.

There are three types of control flow paradigms: procedural, event-driven, and threaded. Our system uses the event-driven and threaded paradigms for control flow. We are using event-driven for its flexibility. Basically, the system responds to the actions (events) that the user submits. This is good practice since resources aren't being wasted. Only when there is an input from the user, will the system respond accordingly with the appropriate service or response.

The application is deployed as a WAR (Web Archive) to the server. This latter runs the application with all the required dependencies. The server, where the application runs, stays on. The user connects to the server via a client to access the remote resources of the server that renders it in a web browser.

We mentioned earlier that the system uses the threaded paradigm in addition to the event-driven one. This is because the system is intended to be used by many different users, each having his or her own session. Any user interacting with the same database and data must run on a different thread. Here, multi-threading would enable the system to be accessible to many users and ensure security and efficiency.

1.7 Boundary Conditions

Prerequisites

Certain conditions must be met before one can effectively interact with the system. These conditions include first, and foremost a physical connection to the network (wired or wireless). The network connection allows the client machine to request services to the remote server.

The client machine also needs to have a compatible web browser given that the application is a web application.

Start-up login

The user starts the application in his/her machine by going through the login and authentication system. The user must provide a valid username and password in order to access the system. The login screen is a subsystem to the main application. The main application “career Tracker” only start upon successful login. The office of the dean determines who is granted access to the system.

Configuration

The user can change some information about their account by clicking on the Profile tab. Here, he/she can easily change their password, security questions, etc. Other options are directly managed by the School of Natural Sciences.

Shutdown

First, the user shuts the system down by logging out of the application, and thus terminating their current session. The system also shuts down when the user closes the web browser that they are currently logged in. And finally, the system shuts itself down after a moment of inactivity from the user.

Exceptions

Here we explore how the system reacts to exceptions that can occur outside the application itself.

Loss of network connection

In case of loss of network connection, the user is notified by the system that the connection to the network or internet has been lost. The user should check the physical or wireless connection to fix and/or restore any broken connection. In case of slower internet speed, the user is also notified, in which case they should get closer to the access point if they were on wireless. Once the network issue has been fixed, the user continues with his/her session.

If the connection loss occurred for just a fraction of seconds, then is restored, the user isn't notified at all. The system simply resumes its services and operates normally.

Wrong file

If the user provides a file with a format that is not compatible with the system, they get a notification with the appropriate message. Likewise, if the file the user is trying to upload is corrupted, they would get a notification with the proper message (Invalid file format, could not read file, etc.). In this case, nothing would be persisted to the database.

Key Personnel Information

William Ntumba – Development Team, Team Leader

Scott Shrout – Development Team

Anthony Freitas – Development Team

Dana Hope – Project Sponsor

Individual Contributions

Scott Shrout

1.1 Overview

1.2 Subsystem Decomposition

1.3 Hardware/Software Mapping

Anthony Freitas

1.4 Persistent Data Management

1.5 Access Control and Security

William Ntumba

1.6 Global Software Control

1.7 Boundary Conditions