



USER'S MANUAL

THIS SOFTWARE IS NOT TO BE USED FOR
UNIVERSITY MANAGEMENT PURPOSES



CSCI-A360

SPRING 2022

ADAM BOESE - ONAN CHEW - JOSH HOLM - AUSTIN WALTER

TABLE OF CONTENTS

PREFACE	<u>2</u>
GETTING STARTED	<u>4</u>
User role selection	<u>5</u>
PROFESSOR DATABASE	
What is the Professor Database?	<u>8</u>
Searching the Database	<u>8</u>
Adding a new Professor	<u>9</u>
Modifying an Existing Professor	<u>10</u>
Deleting a Professor	<u>11</u>
Managing Classes and TAs	<u>12</u>
TEACHER'S ASSISTANT DATABASE	
What is the Teacher's Assistant Database?	<u>15</u>
Searching the Database	<u>15</u>
Adding a new Teacher's Assistant	<u>16</u>
Modifying an Existing Teacher's Assistant	<u>17</u>
Deleting a Teacher's Assistant	<u>18</u>
Managing Classes	<u>19</u>
STAFF DATABASE	
What is the Staff Database?	<u>21</u>
Searching the Database	<u>21</u>
Adding new Staff	<u>22</u>
Modifying Existing Staff	<u>23</u>
Deleting Staff	<u>24</u>
STUDENT DATABASE	
What is the Student Database?	<u>26</u>
Searching the Database	<u>26</u>
Adding a new Student	<u>27</u>
Modifying an Existing Student	<u>28</u>
Deleting a Student	<u>29</u>
Managing Grades	<u>30</u>
APPENDIX	
i. Sample Personnel	<u>33</u>
ii. Class diagram	<u>34</u>
iii. Data dictionary	<u>35</u>
iv. File format	<u>36</u>
v. Use Case Diagram	<u>40</u>
vi. Test cases	<u>48</u>
vii. About the Dev Team	<u>52</u>

PREFACE

University Database Management System Overview

The UofSC Aiken Database Management System is an application designed to aid in the administration and organization of University personnel. Staff, Professors, Teacher's Assistants (TA), and Students may all be managed through this software system. Additionally, course information and student grading can be tracked and stored in the system database.

Application Audience

This program is designed for education administrators and staff, to include University Professors and Teacher's Assistants. The University Database Management System serves to computerize record keeping and personnel organization aimed at reducing manpower and time consumption through a simplified interface.

User Manual Organization

The User Manual is divided into sections covering each functional aspect and use case for the application. After a brief section outlining installation and starting the application, detailed explanations for program processes related to operator profiles is discussed. Finally, the appendix includes comprehensive design documentation serving as the technical framework for the University Database Management System.

- **Getting Started:** Outlines System Requirements and instructions for running the application. An overview of each user role and their functional privileges within the application is discussed.
- **Professor Database:** Provides detailed instructions for the management of Professors - to include adding, modifying, and deleting Professors. Associating Professors with Teacher's Assistants, as well as previously taught courses and current courses are covered.
- **Teacher's Assistant Database:** Provides detailed instructions for the management of Teacher's Assistants (TAs) - to include adding, modifying, and deleting Teacher's Assistants. Associating TAs with previously assisted courses and courses currently assisting is covered.
- **Staff Database:** Provides detailed instructions for the management of University Staff - to include adding, modifying, and deleting staff personnel.

- **Student Database:** Provides detailed instructions for the management of Students - to include adding, modifying, and deleting Students - as well as associating students with courses and overseeing grade management.
- **Appendix:** This section includes sample personnel data and attributes, and also summarizes design processes involved in the development of the University Database Management System application.

Using this User Manual

If utilizing this User's Manual in PDF format, note that clickable links in the following format, [Sample](#), allow you to jump to the relevant section described in the text. For best results, refer to the PDF version of this User's Manual

Feedback

If you would like to contact the development team for licensing purposes, to report bugs, or suggest functionality enhancements - our team can be reached at the following addresses:

Adam Boese	ajboese@usca.edu
Onan Chew	ochew@usca.edu
Josh Holm	jholmjr@usca.edu
Austin Walter	austinwa.ws@usca.edu

GETTING STARTED

System Requirements

University Database Management System was developed with *Java jdk-15.0.1* and compiled into a JAR (Java ARchive) used to aggregate many Java class files and associated metadata and resources into one file for distribution. The UniDB.jar file only requires an installed Java runtime environment, available from <https://java.com/en/download>, and any modern OS. If you are unsure of your computer system's operating environment, refer to your computer's manual, the manufacturer's website, or visit <https://whatsmyos.com>. Alternatively, the source files can be imported into an IDE of your choice, such as Eclipse <https://www.eclipse.org/downloads/>.

University Database Management System is GUI based, meaning it is designed using graphical features such as windows, menus, dialog boxes, and features that make the application easy to use.

Running the Application

- **UniDB.jar:** Double-click the UniDB.jar file.
- **Eclipse:** Start a new Java Project and import the sources files into your Project. Access the 'gui' folder, then open and run the **Login.java** file.

Before proceeding, please read the following *End User License Agreement* and verbally affirm or deny into an audio capture service of your choosing. Send said recording to any member of the [development team](#) prior to utilizing University Database Management System.

THIS SOFTWARE IS NOT TO BE USED FOR
UNIVERSITY MANAGEMENT PURPOSES

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

User Role Selection

On the initial application screen, you will be prompted to select your *USER ROLE*, each granting specific functional permissions within the application. Observe below for an overview of available user roles and capabilities, and select accordingly:

→ **Professor:**

- Add student grades into the University Database Management System.
- Modify the grades of an existing student in the system.
- Display the grades of an existing student in the system.
- Delete the grades of an existing student in the system.

→ **Professor Admin:**

- All permissions encompassed in the Professor Role, plus the following privileges:
 - Add a new professor.
 - Modify any information about an existing professor.
 - Delete a professor.
 - Display the information about a professor

→ **Teacher's Assistant (TA):**

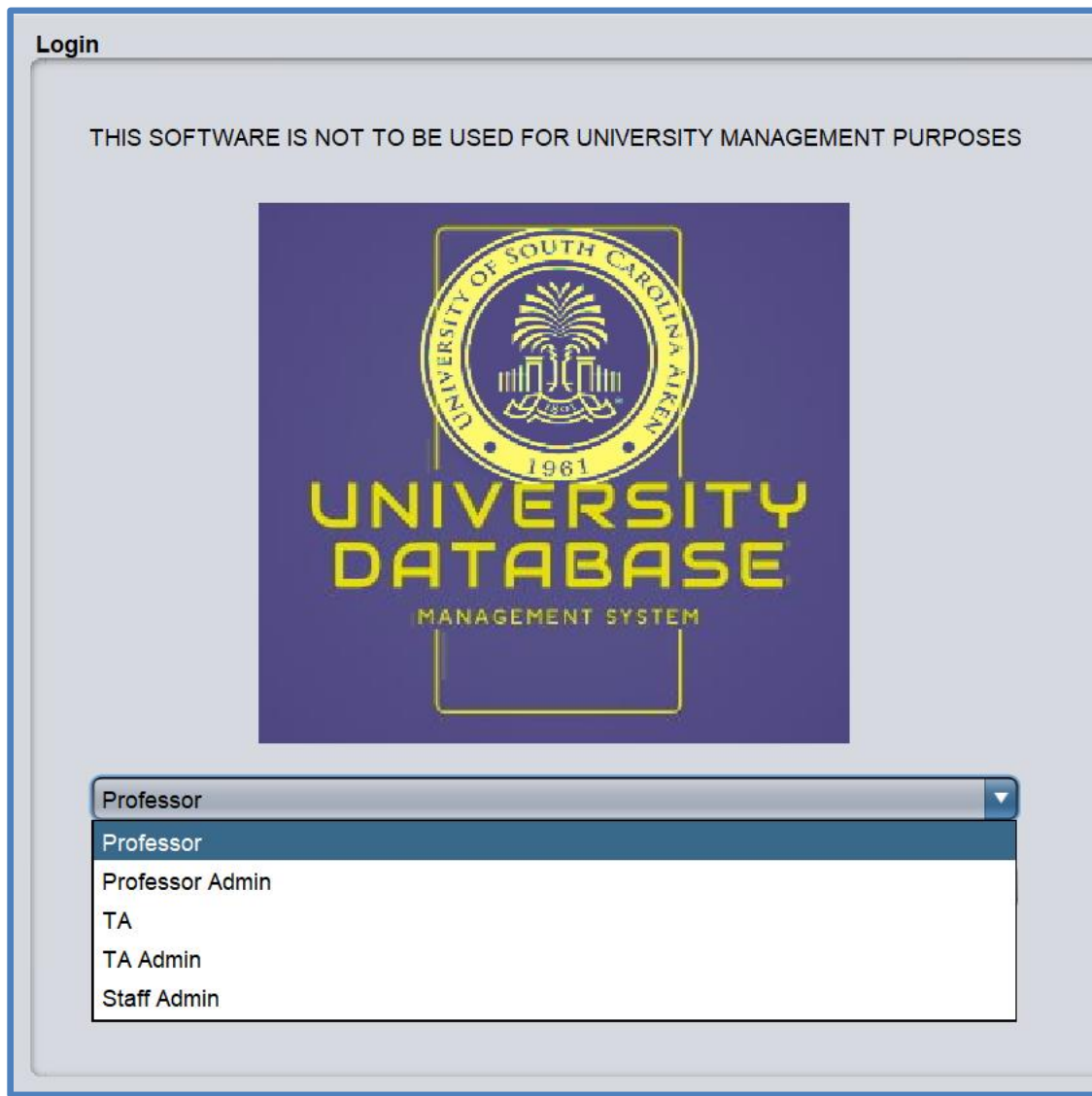
- Add student grades into the University Database Management System.
- Modify the grades of an existing student in the system.
- Display the grades of an existing student in the system.
- Delete the grades of an existing student in the system.

→ **Teacher's Assistant (TA) Admin:**

- All permissions encompassed in the TA Role, plus the following privileges:
 - Add a new TA.
 - Modify any information about an existing TA.
 - Delete a TA.
 - Display the information about a TA.

→ **Staff Member Admin:**

- Add a new staff member.
- Modify any information about an existing staff member.
- Delete a staff member.
- Display the information about a staff member.



Login Screen Instructions

1. Choose desired role by clicking the down arrow adjacent to the selection box and left-clicking the preferred selection.
2. Left-click 'Sign in' to enter the application

✓ Quick Tip:

- To Exit the application at any time, select the 'X' at the top-right corner of the application window.



PROFESSOR DATABASE

What is the Professor Database?

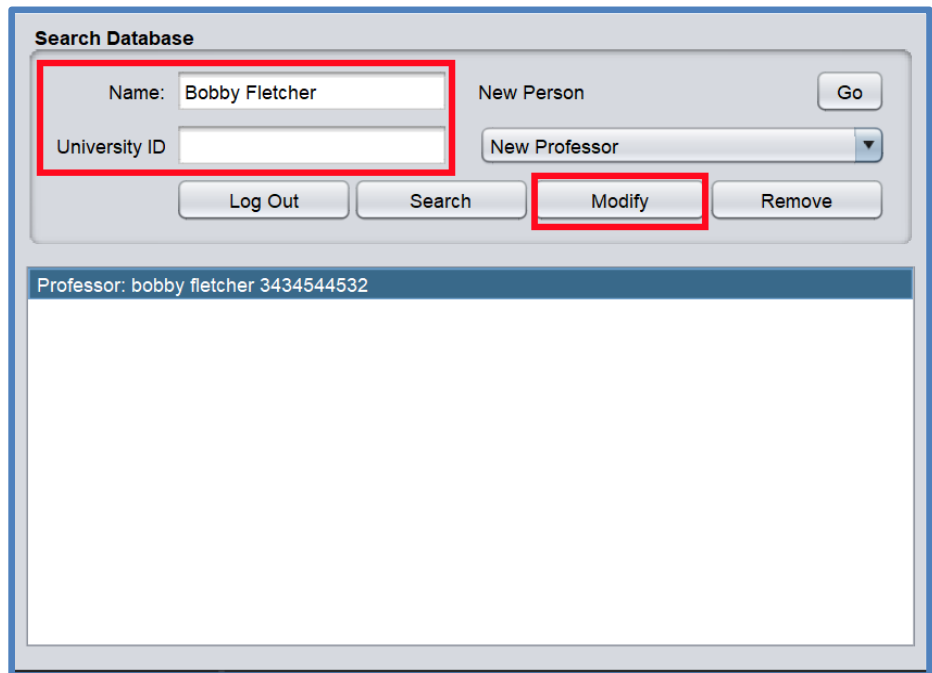
The Professor Database is a listing of instructors teaching at an institution of higher learning. Each Professor is identified by their first and last name, date of birth, university ID, and department. Additionally, Professors are associated with courses previously taught, actively teaching, and Teacher's Assistants (TA) currently assigned to them. Each of the aforementioned attributes associated with Professor may be managed through this interface.

The corresponding data input into the application for each Professor is stored in a text file, Professor.txt. Each row in the file represents a different teacher. Each column per individual row represents information (attributes) pertaining to a single teacher, for example: name, departments, courses teaching, TAs, etc.

In order to add, delete, or make modifications to Professors in the database, you must choose the Professor Admin user role.

Searching the Database

After logging in to the application via your chosen role (Professor Admin), you are presented with the **Search Database** window (see Fig P.1 below). In the **Search Database** window, you may search for the desired Professor by Name or University ID.



The screenshot shows a web application window titled "Search Database". It contains two input fields: "Name:" with the text "Bobby Fletcher" and "University ID" which is empty. To the right of these fields are buttons for "New Person" and "Go". Below the input fields are four buttons: "Log Out", "Search", "Modify", and "Remove". The "Search" button is highlighted with a red box. Below the buttons is a text box displaying "Professor: bobby fletcher 3434544532".

1. Enter the name or university ID for the desired professor.
2. Left-click the **Search** button to view results in the text box.

3. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window for viewing (see Fig P.2 below).

Professors TAs Student People/Staff

First Name: bobby

Last Name: fletcher

Birth Date: 07-31-2002

Department: MATH

University ID: 3434544532

Courses

MATH101

Add Course Teaching

Add

Remove

Add Course Taught

Add

Remove

TAs

dbrown

Add Teachers Assistant

dbrown

Add

Apply

P.2

Adding a new Professor

First, log-in to the application by selecting the Professor Admin user role.

1. Enter the name or university ID for the Professor in the applicable fields on the **Search Database** application window to see if the desired person already exists in the database.
2. If not, select **New Professor** by left-clicking the down-arrow adjacent to the selection box under **New Person** in the **Search Database** window (see Fig P.3).
3. Click **Go**

Search Database

Name: Owens

University ID

New Person

Go

New Professor

Log Out Search Modify Remove

No results!

P.3

4. The **Professors** tab in the **People Properties** application window displays blank text fields for all of the assignable attributes for the new Professor.
5. Enter desired values for each field for the new Professor.
 - Please adhere to the following requirements for each field to avoid errors:
 - First Name: Letters only, maximum length of 25 characters
 - Last Name: Letters only, maximum length of 25 characters
 - Birth Date: 10 characters in the following format: MM-DD-YYYY
 - Department: Letters only, maximum length of 20 characters
 - University ID: Letters and numbers, exactly 9 characters
 - If adding Courses and TAs during Professor creation, refer to the [Managing Classes and Assigning TAs](#) section below.
6. Click the **Add** button to insert the new Professor into the database (see Fig P.4 below).

The screenshot shows the 'People Properties' application window with the 'Professors' tab selected. The form is divided into several sections:

- Professor Information:** A group of five text input fields on the left, each with a label: 'First Name' (containing 'Jerry'), 'Last Name' (containing 'Owens'), 'Birth Date' (containing '02-22-1988'), 'Department' (containing 'CSCI'), and 'University ID' (containing 'JQ3847562'). This entire group is enclosed in a red rectangular box.
- Courses:** A section on the right containing two sub-sections:
 - Add Course Teaching:** Includes a text input field, an 'Add' button, and a 'Remove' button.
 - Add Course Taught:** Includes a text input field, an 'Add' button, and a 'Remove' button.
- TAs:** A section at the bottom left containing a text input field, a dropdown menu (currently showing 'iansnow'), and 'Add' and 'Remove' buttons. The 'Add' button in this section is highlighted with a red rectangular box.

The tabs at the top of the window are 'Professors', 'TAs', 'Student', and 'People/Staff'. The label 'P.4' is visible in the bottom right corner of the window frame.

Modifying an Existing Professor

First, log-in to the application by selecting the Professor Admin user role.

1. Enter the name or university ID for the Professor in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window (see Fig P.5).

Search Database

Name: Bobby Fletcher New Person Go

University ID New Professor

Log Out Search **Modify** Remove

Professor: bobby fletcher 3434544532

P.5

Professors TAs Student People/Staff

First Name: bobby

Last Name: fletcher

Birth Date: 07-31-2002

Department: MATH

University ID: 3434544532

Courses

MATH101 Add Course Teaching

 Add

 Remove

 Add Course Taught

 Add

 Remove

TAs

dbrown Add Teachers Assistant

 dbrown

 Add

Apply

- Each editable field for that Professor is presented in the **People Properties** window. Simply left-click the desired field to edit and replace the current value with your desired input. (Refer to [input requirements](#) for each field above)
- For modification of Courses and TAs, refer to the [Managing Classes and Assigning TAs](#) section below.
- Once the desired changes have been made, click the **Apply** button to write the changes to the database.

Deleting a Professor

First, log-in to the application by selecting the Professor Admin user role.

- Enter the name or university ID for the Professor in the relevant fields on the **Search Database** application window.
- Select the desired result, then click the **Remove** button (see Fig P.6).

Managing Classes and Assigning TAs

Log-in to the application by selecting the Professor Admin user role.

1. Enter the name or university ID for the Professor in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify Selected Item** button to have that person's data populate in the corresponding tab in the **People Properties** window
3. **Classes** (see Fig P.7):
 - To associate a new course with the chosen professor, enter the course name in the **Add Course Teaching** field and click **Add**. Select **Apply** to finalize change.
 - Courses must consist of letters and numbers only, maximum of 20 characters.
 - After a class has ended, the course associated with that professor can be removed by clicking the **Remove** button. Select **Apply** to finalize change.
 - The same mechanism applies to **Add Course Taught**

4. **TAs:**

- A list of available Teacher's Assistants that have already been added to the database management system is populated in the TAs section of the **People Properties** window.
- Simply select the desired TA by left-clicking the down-arrow adjacent to the selection box and clicking the TA's name. Click **Add**, then **Apply**. (see Fig P.8).
- To remove a TA, click the desired name, select **Remove**, then **Apply**.

The screenshot shows the 'People Properties' window with the 'TAs' tab selected. The 'First Name' field contains 'bobby', 'Last Name' contains 'fletcher', 'Birth Date' contains '07-31-2002', 'Department' contains 'MATH', and 'University ID' contains '3434544532'. The 'Courses' section has two empty boxes for 'Add Course Teaching' and 'Add Course Taught', each with 'Add' and 'Remove' buttons. The 'TAs' section at the bottom left has a list box containing 'dbrown' and 'aaronmichaels', and a dropdown menu with 'aaronmichaels' selected. To the right of the TAs list is an 'Add Teachers Assistant' button. A large 'Apply' button is on the far right. Red boxes highlight the TAs list, the dropdown menu, and the 'Apply' button.

P.8

✓ **Quick Tip:**

- To return to the Login screen and switch User Roles, just click **Log Out** from the Search Database application window at any time!



TEACHER'S ASSISTANT DATABASE

What is the Teacher's Assistant Database?

The Teacher's Assistant (TA) Database is a listing of TAs assisting Professors at an institution of higher learning. Each Teacher's Assistant is identified by their first and last name, date of birth, university ID, and department. Additionally, Teacher's Assistants are associated with courses previously assisted and actively assisting with. Each of the aforementioned attributes associated with Teacher's Assistant may be managed through this interface.

The corresponding data input into the application for each Teacher's Assistant is stored in a text file, TeacherAssistant.txt. Each row in the file represents a different TA. Each column per individual row represents information (attributes) pertaining to a single TA, for example: name, departments, courses assisting, etc.

In order to add, delete, or make modifications to Teacher's Assistants in the database, you must choose the Teacher's Assistant Admin user role.

Searching the Database

After logging in to the application via your chosen role, you are presented with the **Search Database** application window (see Fig T.1 below). In the **Search Database** window, you may search for the desired Teacher's Assistant by Name or University ID.

The screenshot shows the 'Search Database' application window. It features a search form with two input fields: 'Name: steph' and 'University ID'. To the right of these fields are buttons for 'New Person' and 'Go'. Below the input fields is a dropdown menu labeled 'New TA'. At the bottom of the form area are four buttons: 'Log Out', 'Search', 'Modify', and 'Remove'. The 'Search' and 'Modify' buttons are highlighted with a red box. Below the form area is a text box containing the text 'TeacherAssistant: steph stevens ss882233'.

1. Enter the name or university ID for the desired Teacher's Assistant.
2. Left-click the **Search** button to view results in the text box.

3. Select the desired result and left-click the **Modif** button to have that person's data populate in the corresponding tab in the **People Properties** window for viewing (see Fig T.2 below).

The screenshot shows the 'People Properties' window with the 'TAs' tab selected. On the left, there are input fields for personal information: First Name (steph), Last Name (stevens), Birth Date (01-01-2001), Department (CSCI), and University ID (ss8822333). An 'Apply' button is at the bottom left. On the right, the 'Courses' section contains two lists. The first list has 'CSCI202' and an 'Add Course Assisting' button. The second list has 'CSCI999' and 'CSCI777' with an 'Add Course Assisted' button. Both lists have 'Add' and 'Remove' buttons.

T.2

Adding a new Teacher's Assistant

First, log-in to the application by selecting the Teacher's Assistant Admin user role.

1. Enter the name or university ID for the Teacher's Assistant in the applicable fields on the **Search Database** application window to see if the desired person already exists in the database.
2. If not, select **New TA** by left-clicking the down-arrow adjacent to the selection box under **New Person** in the **Search Database** window (see Fig T.3).
3. Click **Go**

The screenshot shows the 'Search Database' window. At the top, there are input fields for 'Name' (green) and 'University ID'. To the right, there is a 'New Person' dropdown menu with 'New TA' selected, and a 'Go' button. Below these are buttons for 'Log Out', 'Search', 'Modify', and 'Remove'. At the bottom, a red box highlights the text 'No results!' in the results area.

T.3

4. The **TAs** tab in the **People Properties** application window displays blank text fields for all of the assignable attributes for the new Teacher's Assistant.
5. Enter desired values for each field for the new Teacher's Assistant.
 - Please adhere to the following requirements for each field to avoid errors:
 - First Name: Letters only, maximum length of 25 characters
 - Last Name: Letters only, maximum length of 25 characters
 - Birth Date: 10 characters in the following format: MM/DD/YYYY
 - Department: Letters only, maximum length of 20 characters
 - University ID: Letters and numbers, exactly 9 characters
 - If adding Courses during TA creation, refer to the [Managing Classes](#) section below.
6. Click **Apply**, then **Add** button to insert the new TA into the database (see Fig T.4 below).

The screenshot shows the 'People Properties' application window with the 'TAs' tab selected. The form is divided into two main sections: personal information and course management. The personal information section on the left includes fields for First Name (Pat), Last Name (Green), Birth Date (11-22-1999), Department (ENG), and University ID (PG7846265). The course management section on the right, titled 'Courses', contains two rows. Each row has an 'Add Course Assisting' section with a text input (ENG109 and ENG332 respectively) and 'Add'/'Remove' buttons, and an 'Add Course Assisted' section with a text input and 'Add'/'Remove' buttons. At the bottom of the form are 'Apply' and 'Add' buttons. A red box highlights the personal information fields, and another red box highlights the 'Apply' and 'Add' buttons.

T.4

Modifying an Existing Teacher's Assistant

First, log-in to the application by selecting the Teacher's Assistant Admin user role.

1. Enter the name or university ID for the Teacher's Assistant in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window (see Fig T.5).

Search Database

Name: New Person

University ID: New TA

TeacherAssistant: Pat Green PG7846265

T.5

Professors **TAs** Student People/Staff

First Name:

Last Name:

Birth Date:

Department:

University ID:

Courses

ENG109
ENG111

Add Course Assisting

ENG111

ENG332

Add Course Assisted

- Each editable field for that Teacher's Assistant is presented in the **People Properties** window. Simply left-click the desired field to edit and replace the current value with your desired input. (Refer to [input requirements](#) for each field above)
- For modification of Courses, refer to the [Managing Classes](#) section below.
- Once the desired changes have been made, click the **Apply** button to write the changes to the database.

Deleting a Teacher's Assistant

First, log-in to the application by selecting the Teacher's Assistant Admin user role.

1. Enter the name or university ID for the Teacher's Assistant in the relevant fields on the **Search Database** application window.
2. Select the desired result, then select the **Remove** button (see Fig T.6).

Search Database

Name: green New Person Go

University ID New TA

Log Out Search Modify Remove

TeacherAssistant: Pat Green PG7846265

T.6

Managing Classes

Log-in to the application by selecting the TA Admin user role.

1. Enter the name or university ID for the TA in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window
3. **Classes** (see Fig T.7):
 - To associate a new course with the chosen TA, enter the course name in the **Add Course Assisting** field and click **Add**. Select **Apply** to finalize change.
 - Courses must consist of letters and numbers only, max of 20 characters.
 - After a class has ended, the course associated with that TA can be removed by clicking the **Remove** button. Select **Apply** to finalize change.
 - The same mechanism applies to **Add Course Assisted**

Professors TAs Student People/Staff

First Name: Pat

Last Name: Green

Birth Date: 11-22-1999

Department: ENG

University ID: PG7846265

Courses

ENG111 Add Course Assisting

ENG111 Add Remove

ENG332 Add Course Assisted

ENG227 Add Remove

Apply

T.7



STAFF DATABASE

What is the Staff Database?

The Staff Database is a listing of staff members working at an institution of higher learning. Each Staff member is identified by their first and last name, date of birth, university ID, and department. Each of the aforementioned attributes associated with Staff members may be managed through this interface.

The corresponding data input into the application for each Staff is stored in a text file, Person.txt. Each row in the file represents a different Staff member. Each column per individual row represents information (attributes) pertaining to a single Staff, for example: name, department, University ID, etc.

In order to add, delete, or make modifications to Staff members in the database, you must choose the Staff Admin user role.

Searching the Database

After logging in to the application via your chosen role, you are presented the **Search Database** application window (see Fig S.1 below). In the **Search Database** window, you may search for the desired Staff member by Name or University ID.

S.1

1. Enter the name or university ID for the desired Staff.
2. Left-click the **Search** button to view results in the text box.

3. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window for viewing (see Fig S.2 below).

The image shows a window titled 'People Properties' with four tabs: 'Professors', 'TAs', 'Student', and 'People/Staff'. The 'People/Staff' tab is selected. Inside the tab, there are five input fields: 'First Name' with the value 'bob', 'Last Name' with the value 'tyler', 'Birth Date' with the value '07-31-2001', 'Department' with the value 'HR', and 'University ID' with the value '389746238'. Below these fields is a large 'Apply' button.

S.2

Adding a new Staff member

First, log-in to the application by selecting the Staff Admin user role.

1. Enter the name or university ID for the Staff in the applicable fields on the **Search Database** application window to see if the desired person already exists in the database.
2. If not, select **New Staff** by left-clicking the down-arrow adjacent to the selection box under **New Person** in the **Search Database** window (see Fig S.3).
3. Click **Go**

The image shows a window titled 'Search Database'. It has two input fields: 'Name' with the value 'aaron' and 'University ID' which is empty. To the right of these fields is a 'New Person' section containing a dropdown menu with 'New Person/staff' selected and a 'Go' button. Below the input fields are four buttons: 'Log Out', 'Search', 'Modify', and 'Remove'. At the bottom of the window, there is a large white box containing the text 'No results!'.

S.3

4. The **People/Staff** tab in the **People Properties** application window displays blank text fields for all of the assignable attributes for the new Staff member.
5. Enter desired values for each field for the new Staff.
 - Please adhere to the following requirements for each field to avoid errors:
 - First Name: Letters only, maximum length of 25 characters
 - Last Name: Letters only, maximum length of 25 characters
 - Birth Date: 10 characters in the following format: MM/DD/YYYY
 - Department: Letters only, maximum length of 20 characters
 - University ID: Letters and numbers, exactly 9 characters
6. Click **Apply**, then **add** button to insert new Staff into the database (see Fig S.4 below).

The screenshot shows the 'People Properties' application window with the 'People/Staff' tab selected. The form contains the following fields and values:

Field	Value
First Name	Zach
Last Name	Steen
Birth Date	06-16-1967
Department	Finance
University ID	ZS8495032

At the bottom of the form, there are two buttons: 'Apply' and 'Add'. Both buttons are highlighted with red boxes. The 'Add' button is located below the 'Apply' button.

S.4

Modifying an Existing Staff member

First, log-in to the application by selecting the Staff Admin user role.

1. Enter the name or university ID for the Staff in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window (see Fig S.5).

The screenshot shows the 'Search Database' application window. The form contains the following fields and values:

Field	Value
Name	chew
University ID	

At the bottom of the form, there are four buttons: 'Log Out', 'Search', 'Modify', and 'Remove'. The 'Modify' button is highlighted with a red box. Below the buttons, there is a result display area showing 'Person: onan chew OC3210839'. This result is also highlighted with a red box.

S.5

- Each editable field for that Staff member is presented in the **People Properties** window. Simply left-click the desired field to edit and replace the current value with your desired input. (Refer to [input requirements](#) for each field above)
- Once the desired changes have been made, click the **Apply** button to write the changes to the database. (see Fig S.5a).

Professors TAs Student **People/Staff**

First Name: onan

Last Name: chew

Birth Date: 01-07-2000

Department: science

University ID: OC3210839

Apply

S.5a

Deleting a Staff member

First, log-in to the application by selecting the Staff Admin user role.

- Enter the name or university ID for the Staff in the relevant fields on the **Search Database** application window.
- Select the desired result and left-click the **Modify Selected Item** button to have that person's data populate in the corresponding tab in the **People Properties** window
- On the **People Properties** window, select the **Remove** button (see Fig S.6).

Search Database

Name: wallace

University ID

New Person

Go

New Person/staff

Log Out Search Modify Remove

Person: austin wallace 343454454

S.6



STUDENT DATABASE

What is the Student Database?

The Student Database is a listing of students taking courses at an institution of higher learning. Each Student is identified by their first and last name, date of birth, university ID, and department. Additionally, Students are associated with courses taken which includes the course name, semester, and corresponding letter & number grade. Each of the aforementioned attributes associated with Students may be managed through this interface.

Data input into the application for each Student is stored in a text file, Student.txt. Each row in the file represents a different Student. Each column per individual row represents information (attributes) pertaining to a single student, for example: name, birth date, courses taken, etc.

In order to add, delete, or make modifications to Students in the database, you must choose the Professor, Professor Admin, TA, or TA Admin user role.

Searching the Database

After logging in to the application via your chosen role, you are presented with the **Search Database** application window (see Fig Z.1 below). In the **Search Database** window, you may search for the desired Student by Name or University ID.

Search Database

Name: josh New Person Go

University ID New Student ▼

Log Out Search Modify Remove

Student: josh holm pff111111

Z.1

1. Enter the name or university ID for the desired Student.
2. Left-click the **Search** button to view results in the text box.
3. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window for viewing (see Fig Z.2 below).

Adding a new Student

First, log-in to the application by selecting your desired user role.

1. Enter the name or university ID for the Student in the applicable fields on the **Search Database** application window to see if the desired person already exists in the database.
2. If not, select **New Student** by left-clicking the down-arrow adjacent to the selection box under **New Person** in the **Search Database** window (see Fig Z.3).
3. Click **Go**

Z.3

4. The **Student** tab in the **People Properties** application window displays blank text fields for all of the assignable attributes for the new Student.
5. Enter desired values for each field for the new Student.
 - Please adhere to the following requirements for each field to avoid errors:
 - First Name: Letters only, maximum length of 25 characters
 - Last Name: Letters only, maximum length of 25 characters
 - Birth Date: 10 characters in the following format: MM/DD/YYYY
 - Department: Letters only, maximum length of 20 characters
 - University ID: Letters and numbers, exactly 9 characters
 - If adding Courses/Grades during Student creation, refer to the [Managing Grades](#) section below.
6. Click **Apply**, then **Add** to insert the new Student into the database (see Fig Z.4 below).

Professors TAs **Student** People/Staff

First Name: brent

Last Name: thompson

Birth Date: 04-15-2002

Department: CSCI

University ID: BT6475843

Courses Taken

Course Name

Course Grade

Course Semester ID

Remove

Add

Apply

Z.4

Modifying an Existing Student

First, log-in to the application by selecting your desired user role.

1. Enter the name or university ID for the Student in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify** button to have that person's data populate in the corresponding tab in the **People Properties** window (see Fig Z.5).

Search Database

Name: brent

University ID

New Person

Go

New Student

Log Out Search **Modify** Remove

Student: brent thompson BT6475843

Z.5

- Each editable field for that Student is presented in the **People Properties** window. Simply left-click the desired field to edit and replace the current value with your desired input. (Refer to [input requirements](#) for each field above)
- For modification of Grades, refer to the [Managing Grades](#) section below.
- Once the desired changes have been made, click the **Apply** button to write the changes to the database (see Fig Z.5a).

Deleting a Student

First, log-in to the application by selecting the desired user role.

- Enter the name or university ID for the Student in the relevant fields on the **Search Database** application window.
- Select the desired result and left-click the **Modify Selected Item** button to have that person's data populate in the corresponding tab in the **People Properties** window
- On the **People Properties** window, select the **Remove** button (see Fig Z.6).

Managing Grades

Log-in to the application by selecting your desired user role.

1. Enter the name or university ID for the Student in the relevant fields on the **Search Database** application window.
2. Select the desired result and left-click the **Modify Selected Item** button to have that person's data populate in the corresponding tab in the **People Properties** window
3. **Courses Taken:**
 - In the **Courses Taken** section of the **Student** tab in **People Properties**, three fields are present to enter the **Course Name**, **Semester**, and **Course Grade** for a course the student has taken.
 - Please adhere to the following input conventions to avoid errors:
 - Course: Letters and numbers only, maximum 8 characters
 - Semester: Letters and numbers only, maximum 12 characters
 - Grade: Numbers only, between 0 and 100.
 - To enter a **New Course and Grades**, simply fill the blanks fields and select **Add** then **Apply** (see Fig Z.7).

The screenshot shows the 'People Properties' window with the 'Student' tab selected. The 'Courses Taken' section is highlighted with a red box. It contains a list box with the text 'CourseID: CSCI502 SemesterID: SPRING22'. To the right of the list box are three input fields: 'Course Name' (CSCI502), 'Course Grade' (83), and 'Course Semester ID' (SPRING22). Below these fields are 'Remove' and 'Add' buttons. At the bottom of the window, an 'Apply' button is highlighted with a red box.

Z.7

- To **Delete Grades**, select the course in the **Courses Taken** box to populate the input fields. Select **Remove** then **Apply** (see Fig Z.8).

Courses Taken

CourseID: CSCI502 SemesterID: SPRING22

Course Name
CSCI502

Course Grade
83

Course Semester ID
SPRING22

Remove

Add

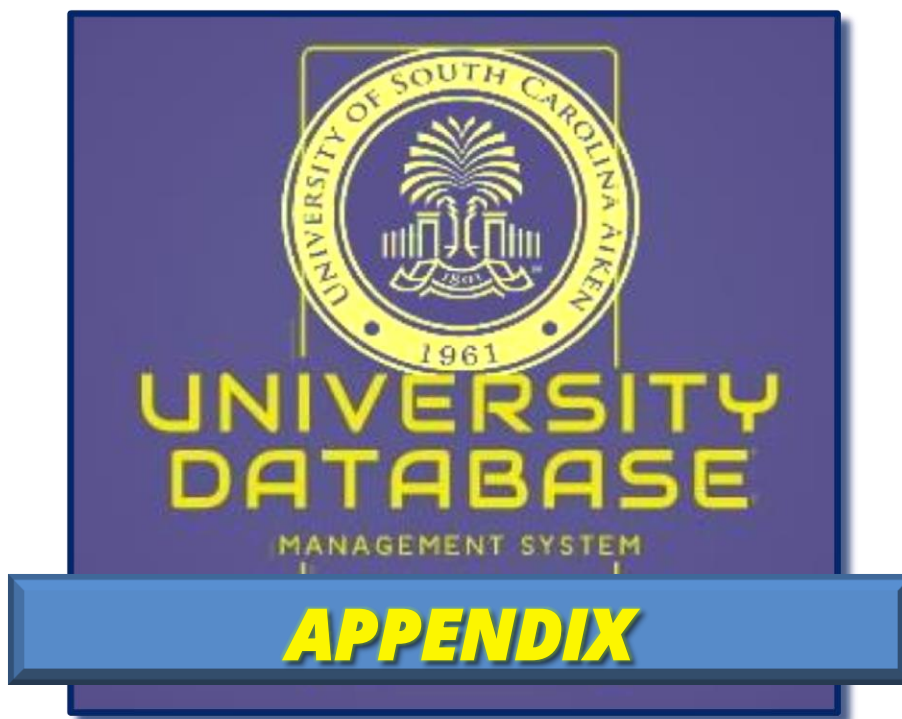
Apply

Z.8

- To **Modify Grades**, select the course in the **Courses Taken** box to populate the input fields. Edit as needed and select **Add**. Then select the original course and choose **Remove**.

✓ **Quick Tip:**

- To return to the Login screen and switch User Roles, just click *Log Out* from the Search Database application window at any time!



APPENDIX

Sample Personnel Data

PROFESSOR

- First Name: *John*
- Last Name: *Brown*
- Birth Date: *03/04/1982*
- Department: *CSCI*
- University ID: *JB3482111*
- Course Teaching: *CSCI101*
- Course Taught: *CSCI211*

TA

- First Name: *Aaron*
- Last Name: *Smiff*
- Birth Date: *02/11/2002*
- Department: *MATH*
- University ID: *AS0211111*
- Course Assisted: *MATH101*
- Course Assisting: *MATH225*

STAFF

- First Name: *Barb*
- Last Name: *Weathers*
- Birth Date: *05/22/1991*
- Department: *Admissions*
- University ID: *BW0522111*

STUDENT

- First Name: *Bobby*
- Last Name: *Green*
- Birth Date: *11/02/2001*
- Department: *HIST*
- University ID: *BG1102111*
- Course ID: *HIST101*
- Semester: *Spring2022*
- Number Grade: *84*

Class diagram



Data dictionary

Admin - an administrative role that affords authenticated users additional database privileges based upon their primary role. *Professors* with admin privileges may add, modify, delete, and display information about other professors. *Teacher's Assistants* may add, modify, delete, and display information about other TA's. General staff may add, modify, delete, and display information about other staff through the administrative role.

Student - Each Student record consists of student's first and last name, courses taken each semester, and associated letter and numerical grade for each course. *Professors* and *Teacher's Assistants* both have the ability to add student records and modify, display, or delete corresponding grades for courses attended.

Professor - a record and user role associated with a professor. Each *Professor* is identified by their first and last name, date of birth, university ID, and department. Each Professor is associated with courses previously taught, actively teaching, and Teacher's Assistants assigned to them. This role may add, modify, display, and delete student grades from the database.

ProfessorManager - a class allowing users to add, modify, delete, and display information relating to *Professor Objects*.

Login - a *graphical user interface* window that prompts for a specified *User Role* selection, through which database privileges and granted dependent upon their primary role.

PeopleProperties – a *multi-tabbed graphical user interface* window for managing the creation, deletion, and modification of a person and their attributes in the management database. Data related to each *Person* record, including TA's, professors, students, and staff may be displayed, entered, modified, or deleted should sufficient privileges be associated with the accessing user role.

Person - a record and user role containing baseline attributes and operations for extended roles. *Person* contains attributes and operations common to *Teacher's Assistant*, *Student*, *Teacher*, *Staff* and admin database records and roles and thus each of these roles is also a Person.

StaffManager - a class allowing users to add, modify, delete, and display information relating to *people objects*.

SearchDataBase - a *graphical user interface* window that permits access to and modification of information in the database. Users may enter search criteria such as name or other unique attribute to access specific *Person* data records for TA's, professors, students, and staff. Selection of a record directs to the *PeopleProperties* GUI window where information may be displayed, entered, modified, or deleted - dependent upon privileges associated with the accessing user.

TeacherAssistantManager - a class allowing users to add, modify, delete, and display information relating to *Teacher's Assistant objects*.

TeacherAssistant- a record and user role associated with a teacher assistant. Each *Teacher's Assistant* is identified by their first and last name, date of birth, university ID, and department. Each Teacher's Assistance is associated with courses previously assisted and currently assisting. This role may add, modify, display, and delete student grades from the database.

File format

Overview:

- There are a total of 4 data files: Professor.txt, TeacherAssistant.txt, Student.txt, Person.txt
- Text file format type.
- Each file will contain rows consisting of a single object of the specified class, i.e. each row in Professor.txt will contain a single Professor.
- Columns for each row represent attributes associated with that object. Row sizes will not be prefixed - as a variable number of attributes exist depending on the object, i.e. one Professor may be teaching more courses than another and thus have additional attributes. Column attributes will be separated by commas.

The following is a description of the data files along with a diagram that represents the data.

Professor.txt

This file holds all existing Professors in our Database. Each row in the file will represent a different Professor. Each column in the row represents information (attributes) pertaining to a single Professor. Professors will be able to add, modify, display, and delete grades for students, however this data will be stored in the Student.txt file. As the number of crsTaught, crsTeaching, and TAs per Professor may vary, these will be stored as sub-arrays but printed in sequence per Professor row.

Professor 1

FirstName	LastName	birthDate	uniID	Department	isStaff	isAdmin	[crsTaught]	[crsTeaching]	[TAs]
-----------	----------	-----------	-------	------------	---------	---------	-------------	---------------	-------

Professor 2

FirstName	LastName	birthDate	uniID	Department	isStaff	isAdmin	[crsTaught]	[crsTeaching]	[TAs]
-----------	----------	-----------	-------	------------	---------	---------	-------------	---------------	-------

.

Professor n

FirstName	LastName	birthDate	uniID	Department	isStaff	isAdmin	[crsTaught]	[crsTeaching]	[TAs]
-----------	----------	-----------	-------	------------	---------	---------	-------------	---------------	-------

firstName: This is the first name of Professor. It is a string type with a predefined maximum size of 25 characters.

lastName: This is the last name of Professor. It is a string type with a predefined maximum size of 25 characters.

birthDate: This is the date of birth for the Professor in the following format: MM-DD-YYYY - i.e. (02-23-1987). It is a string type with predefined max size of 10 characters.

uniID: This is the Professor's ID for the university and is unique to one Professor. It is a string type with predefined max size of 9 characters.

department: This is the abbreviated department the Professor is in. It is a string type with a predefined maximum size of 20.

isAdmin: This will determine whether the user is an admin. If the user has a Boolean of true, then the user has admin permissions. Admin permissions will allow the user to add, display, modify and delete Professors.

isStaff: This will determine whether the user is staff. If the user has a Boolean of true, then the user is a staff member.

crsTaught: This is a list of courses the Professor has taught contained in an array. Each element is a string type with a predefined maximum size of 20.

crsTeaching: This is a list of courses the Professor is currently teaching contained in an array. Each element is a string type with a predefined maximum size of 20.

TAs: This is a list of Teacher assistants under the Professor contained in an array. Each element is a string type with a predefined maximum size of 20.

TeacherAssistant.txt

This file holds all existing teacher assistants in our Database. Each row in the file will represent a different teacher assistant. Each column in the row represents information (attributes) pertaining to each teacher assistant. TeacherAssistants will be able to add, modify, display, and delete grades for students, however this data will be stored in the Student.txt file. As the number of crsAssist and crsAssisting per TA may vary, these will be stored as sub-arrays but printed in sequence per TeacherAssistant row.

TeacherAssistant 1

firstName	lastName	birthDate	uniID	isStaff	isAdmin	department	[crsAssist]	[crsAssisting]
-----------	----------	-----------	-------	---------	---------	------------	-------------	----------------

TeacherAssistant 2

firstName	lastName	birthDate	uniID	isStaff	isAdmin	department	[crsAssist]	[crsAssisting]
-----------	----------	-----------	-------	---------	---------	------------	-------------	----------------

•
•
•

TeacherAssistant n

firstName	lastName	birthDate	uniID	isStaff	isAdmin	department	[crsAssist]	[crsAssisting]
-----------	----------	-----------	-------	---------	---------	------------	-------------	----------------

firstName: This is the first name of the teacher assistant. It is a string type with a predefined maximum size of 25 characters.

lastName: This is the last name of the teacher assistant. It is a string type with a predefined maximum size of 25 characters.

birthDate: This is the date of birth for the teacher assistant teacher in the following format: MM-DD-YYYY - i.e. (02-23-1987). It is a string type with predefined max size of 10 characters.

uniID: This is the teacher assistant's ID for the university and is unique to one teacher assistant. It is a string type with predefined max size of 9 characters.

isStaff: This will determine whether the user is staff. If the user has a Boolean of true, then they are a staff member.

isAdmin: This will determine whether the user is an admin. If the user has a Boolean of true, then the user has admin permissions. Admin permissions will allow the user to add, display, modify and delete teacher assistants.

department: This is the abbreviated department the teacher assistant is in. It is a string type with a predefined maximum size of 20.

crsAssisted: This is a list of courses the TA has assisted with contained in an array. Each element is a string type with a predefined maximum size of 20.

crsAssisting: This is a list of courses the TA is currently assisting contained in an array. Each element is a string type with a predefined maximum size of 20.

Student.txt

This file holds all existing students in our database. Each row in the file will represent a different student. This will hold the grades of students input by Professors. *crsID*, *semesID*, *numGrade* and *letGrade* will be implemented in a set and depending on how many courses a student has taken will have either 1 or more sets of data in it. The first five columns in the row represent information regarding students and the last column represents the grades the student has obtained.

Student 1

firstName	lastName	birthDate	uniID	isStaff	isAdin	department	[crsID	semesID	numGrade	letGrade]
							Course info			

Student 2

firstName	lastName	birthDate	uniID	isStaff	isAdin	department	[crsID	semesID	numGrade	letGrade]
							Course info			

•
•
•

Student n

firstName	lastName	birthDate	uniID	isStaff	isAdin	department	[crsID	semesID	numGrade	letGrade]
							Course info			

firstName: This is the first name of the student. It is a string type with a predefined max size of 25 characters.

lastName: This is the last name of the student. It is a string type with a max predefined size of 25 characters.

birthDate: This is the date of birth for the student in the following format: MM-DD-YYYY - i.e. (02-23-1987). It is a string type with predefined max size of 10 characters.

uniID: This is the student ID for the university and is unique to one student. It is a string type with predefined max size of 9 characters.

isStaff: This will determine whether the user is staff. If the user has a Boolean of true, then the user is a staff member.

isAdmin: This will determine whether the user is an admin. If the user has a Boolean of true, then the user has admin permissions. There are no default student admin privileges as this value will remain false for all students.

department: This is the abbreviated department associated with the student's major. It is a string type with a predefined maximum size of 20.

crsID: This is the ID for a course a student has taken. This will be paired with *semesID*, *numGrade*, and *letGrade* to give a student a grade. It is a string type with a defined max size of 8

semesID: This is the semester for the course taken. This will be paired with *crsID*, *numGrade*, and *letGrade* to give a student a grade. It is a string type with a predefined max size of 12.

numGrade: This is the number grade the student acquired in courses based off the *crsID* and *semesID*. This will be paired with *crsID*, *semesID*, and *numGrade* to give the student a grade. It is an integer type with a predefined max size of 3.

letGrade: This is the letter grade the student acquired in a course based off the *crsID* and *semesID*. This will be paired with *crsID*, *semesID*, and *letGrade* to give the student a grade. It is a string type with a defined max size of 1.

Person.txt

This file holds all staff members in our Database. Each staff member is a member of the Person class but identified as a staff member by the *isStaff* Boolean attribute. Each row in the file will represent a different staff member. Each column in the row represents information regarding staff.

Person/Staff 1

firstName	lastName	birthDate	uniID	isStaff	isAdmin	department
-----------	----------	-----------	-------	---------	---------	------------

Person/Staff 2

firstName	lastName	birthDate	uniID	isStaff	isAdmin	department
-----------	----------	-----------	-------	---------	---------	------------

•
•
•

Person/Staff n

firstName	lastName	birthDate	uniID	isStaff	isAdmin	department
-----------	----------	-----------	-------	---------	---------	------------

firstName: This is the first name of the staff member. It is a string type with a max predefined size of 25 characters.

lastName: This is the last name of a staff member. It is a string type with a max predefined size of 25 characters.

birthDate: This is the date of birth for the staff in the following format: MM-DD-YYYY - i.e. (02-23-1987). It is a string type with predefined max size of 10 characters.

uniID: This is the staff's ID for the university and is unique to staff members. It is a string type with predefined max size of 9 characters.

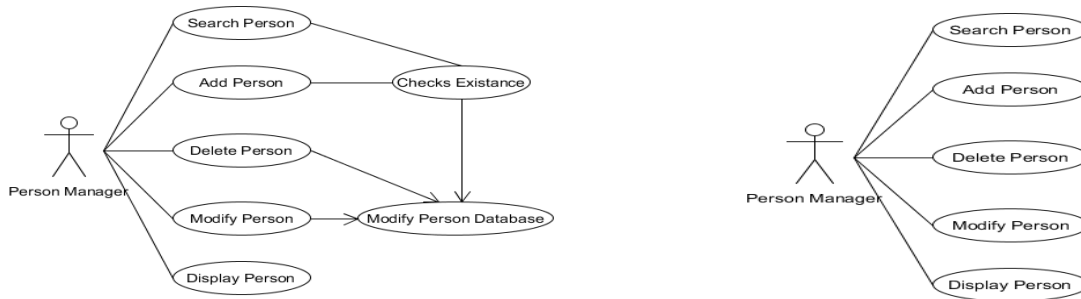
isStaff: This will determine whether the user is staff. If the user has a Boolean of true, then they are a staff member.

isAdmin: This will determine whether the user is an admin. If the user has a Boolean of true, then the user has admin permissions. Admin permissions will allow the user to add, display, modify and delete people/staff.

department: This is the abbreviated department the staff member belongs to. It is a string type with a predefined maximum size of 20.

Use Case Diagrams

Staff Manager Use-case diagram



System	University Database
Use Case	Add person
Actors	Person Manager
Description	The person Manager, using the add Person method and a provided person object, adds a person to the person database. Before the person can be added to the database, they are first checked to ensure they do not already exist within the provided context.
Stimulus	The user has queried for a person object to be added to the database with valid attributes
Response	If the Person does exist, the method returns false to the user - displaying a message relating to the error. If the Person does not exist, the method returns true and then adds the person to the person database
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking the existence of the person prevents inadvertent overwriting, or duplication of data.

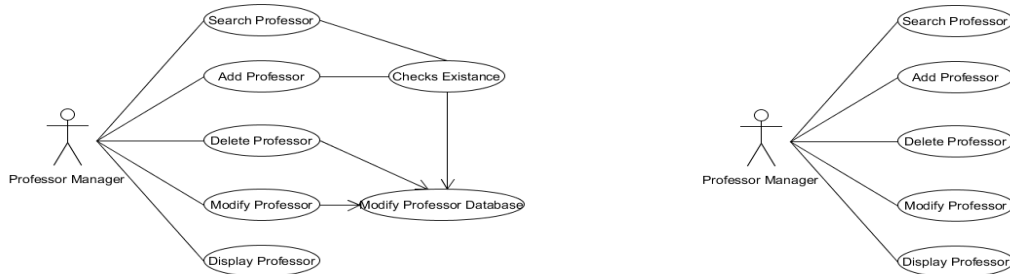
System	University Database
Use Case	Search Person
Actors	Person Manager
Description	The Person Manager, using the search method and provided search terms or attributes, searches the database for people with data that matches with object attributes sent as parameters
Stimulus	The user has queried for a search with valid attribute data
Response	A list of Person objects with matching data attributes is returned to the querying user. If no person exists, the method will return null
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking the existence of the person prevents inadvertent overwriting, or duplication of data.

System	University Database
Use Case	Delete Person
Actors	Person Manager
Description	The Person Manager, using the delete Person method and the already provided person object from the search method, deletes the Person from the person database
Stimulus	The user has queried for a person to be deleted
Response	If the Person deletion fails, the method returns false. If the Person deletion is successful, the method returns true and the relevant object removed from the database
Comments	Because the Person must already be displayed for the user to have the option to delete, the Person the method does not need to check if they exist

System	University Database
Use Case	Display Person
Actors	Person Manager
Description	The Person Manager, using the Display Person method and the response (the person object) from the search method, displays all available information pertaining to the user.
Stimulus	The user has chosen a Person object returned by the Search person method to be displayed
Response	Displays all the information on the person chosen by the user
Comments	The key use of the display Person method is to display ALL information on a person object in the GUI for the user utilizing the object returned by the search Person method.

System	University Database
Use Case	Modify Person
Actors	Person Manager
Description	The Person Manager, using the modify Person method and the Person object provided by the search Person method, modifies specific values of a Person within the Person database by over-writing the old object entry with the new object.
Stimulus	The user queried for a person to be modified with new valid attribute data
Response	If the Person is not modified, the method returns false. If the Person is modified, the method returns true
Comments	Because the Person must already be displayed for the user to have the option to modify, the method does not need to check if they already exist in the database

Professor Manager Use-case diagram



System	University Database
Use Case	Add Professor
Actors	Professor Manager
Description	The Professor Manager, using the add Professor method and a provided Professor object, adds a Professor to the Professor database. Before the Professor can be added to the database, they are first checked to make sure they do not already exist within the provided context
Stimulus	The user has queried for a Professor object to be added to the database with valid attributes
Response	If the Professor does exist, the method returns false to the user - displaying a message relating to the error. If the Professor does not exist, the method returns true and adds the Professor to the Professor database
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking the existence of the Professor prevents inadvertent overwriting, or duplication of data.

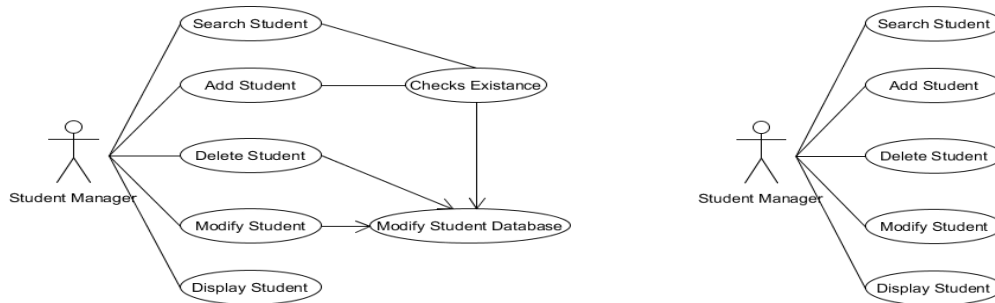
System	University Database
Use Case	Search Professor
Actors	Professor Manager
Description	The Professor Manager, using the search method and provided search terms or attributes, searches the database for People with data that match all search parameters
Stimulus	The user has queried for a search with valid attribute data
Response	A list of Professor objects with matching data attributes are returned to the querying user. If no professor exists, the method returns null
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking the existence of the Professor prevents inadvertent overwriting, or duplication of data.

System	University Database
Use Case	Delete Professor
Actors	Professor Manager
Description	The Professor Manager, using the delete Professor method and the already provided Professor object from the search method, deletes the Professor from the Professor database
Stimulus	The user has queried for a professor to be deleted
Response	If the Professor deletion fails, the method returns false. If the Professor is deleted, the method returns true and object is removed from the database
Comments	Because the Professor must already be displayed for the user to have the option to delete the Professor, the method does not need to check if they exist

System	University Database
Use Case	Display Professor
Actors	Professor Manager
Description	The Professor Manager, using the Display Professor method and the response (the Professor object) from the search method, displays all available information from the user.
Stimulus	The user has chosen a specific Professor from the Professor objects returned by the Search Professor method to be displayed
Response	Displays all information pertaining to the Professor chosen by the user
Comments	The key use of the display Professor method is to display ALL information pertaining to a Professor object within the GUI - after user selection following the search Professor method.

System	University Database
Use Case	Modify Professor
Actors	Professor Manager
Description	The Professor Manager, using the modify Professor method and the Professor object provided by the search Professor method, modifies specific values of a Professor within the Professor database by over-writing the old data with the new object.
Stimulus	The user has queried for a Professor to be deleted with new valid attribute data
Response	If the Professor is not modified, the method returns false. If the Professor is modified, the method returns true
Comments	Because the Professor must already be displayed for the user to have the option to modify a Professor, the method does not need to check if they exist

Student Manager Use-case diagram



System	University Database
Use Case	Add Student
Actors	Student Manager
Description	The Student Manager, using the add Student method and a provided Student object, adds a Student to the Student database. Before the Student can be added to the database, they are first checked to make sure they do not already exist within the provided context.
Stimulus	The user has queried for a Student object to be added to the database with valid attributes
Response	If the Student does exist, the method returns false to the user, displaying a message relating to the error. If the Student does not exist, the method returns true and then adds the Student to the Student database
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking for the existence of the Student prevents inadvertent overwriting, or duplication of data.

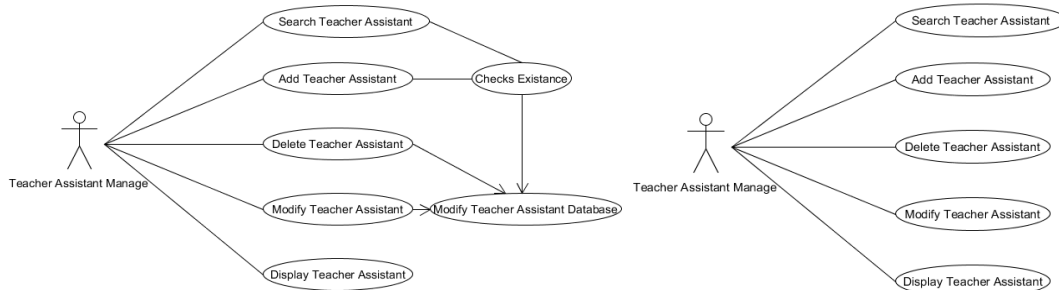
System	University Database
Use Case	Search Student
Actors	Student Manager
Description	The Student Manager, using the search method and provided search terms or attributes searches the database for Students with data that match with all the query parameters
Stimulus	The user has queried for a search with valid attribute data
Response	A list of Student objects with matching data attributes is returned to the querying user. If no matching Student exists, the method will return null
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking for the existence of the Student prevents inadvertent overwriting, or duplication of data.

System	University Database
Use Case	Delete Student
Actors	Student Manager
Description	The Student Manager, using the delete Student method and the already provided Student object from the search method, deletes the Student from the Student database
Stimulus	The user has queried for a Student to be deleted
Response	If the Student deletion fails, the method returns false. If Student deletion is successful, the method returns true and object removed from database
Comments	Because the Student must already be displayed for the user to have the option to delete, the method does not need to check if they exist

System	University Database
Use Case	Display Student
Actors	Student Manager
Description	The Student Manager, using the Display Student method and the response (the Student object) from the search method, displays all available information pertaining to the user.
Stimulus	The user has chosen a Student from the Student objects returned by the Search Student method to be displayed
Response	Displays all the information pertaining to the Student chosen by the user
Comments	The key use of the display Student method is to display ALL information pertaining to a Student object within the GUI - after user selection following the search Student method.

System	University Database
Use Case	Modify Student
Actors	Student Manager
Description	The Student Manager, using the modify Student method and the Student object provided by the search Student method, modifies specific values of a Student within the Student database by over-writing the old data with the new object.
Stimulus	The user has queried for a Student to be deleted with new valid attribute data
Response	If the Student is not modified, the method returns false. If the Student is modified, the method returns true and object over-written with updated attributes
Comments	Because the Student must already be displayed for the user to have the option to modify a Student, the method does not need to check if they exist

Teacher Assistant Manager Use-case diagram



System	University Database
Use Case	Add Teacher Assistant
Actors	Teacher Assistant Manager
Description	The Teacher Assistant Manager, using the add Teacher Assistant method and a provided Teacher Assistant object, adds a Teacher Assistant to the Teacher Assistant database. Before the Teacher Assistant can be added to the database, they are first checked to make sure they do not already exist within the provided context.
Stimulus	The user has queried for a Teacher Assistant object to be added to the database with valid attributes
Response	If the Teacher Assistant does exist, the method returns false to the user, displaying a message relating to the error. If the Teacher Assistant does not exist, the method returns true and then adds the Teacher Assistant to the Teacher Assistant database
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking for the existence of a Teacher Assistant prevents inadvertent overwriting, or duplication of data.

System	University Database
Use Case	Search Teacher Assistant
Actors	Teacher Assistant Manager
Description	The Teacher Assistant Manager, using the search method and provided search terms or attributes, searches the database for Teacher Assistants with data that match with all the data sent as parameters
Stimulus	The user has queried for a search with valid attribute data
Response	A list of Teacher Assistant objects with matching data attributes is returned to the querying user. If no Teacher Assistant exists, the method will return null
Comments	It is important to check errors on the back end even if they may not affect the system directly. Checking for the existence of a Teacher Assistant prevents inadvertent overwriting, or duplication of data.

System	University Database
Use Case	Delete Teacher Assistant
Actors	Teacher Assistant Manager
Description	The Teacher Assistant Manager, using the delete Teacher Assistant method and the already provided Teacher Assistant object from the search method, deletes the Teacher Assistant from the Teacher Assistant database
Stimulus	The user has queried for a Teacher Assistant to be deleted
Response	If the Teacher Assistant is failed to be deleted, the method returns false. If the Teacher Assistant is deleted, the method returns true object deleted from the database
Comments	Because the Teacher Assistant must already be displayed for the user to have the option to delete, the method does not need to check if they exist

System	University Database
Use Case	Display Teacher Assistant
Actors	Teacher Assistant Manager
Description	The Teacher Assistant Manager, using the Display Teacher Assistant method and the response (the Teacher Assistant object) from the search method, displays all available information to the user.
Stimulus	The user has chosen a Teacher Assistant out of the Teacher Assistant objects returned by the Search Teacher Assistant method to be displayed
Response	Displays all the information on the Teacher Assistant chosen by the user
Comments	The key use of the display Teacher Assistant method is to display ALL information pertaining to a Teacher Assistant object within the GUI - after user selection following the search Teacher Assistant method.

System	University Database
Use Case	Modify Teacher Assistant
Actors	Teacher Assistant Manager
Description	The Teacher Assistant Manager, using the modify Teacher Assistant method and the Teacher Assistant object provided by the search Teacher Assistant method, modifies specific values of a Teacher Assistant within the Teacher Assistant database by over writing the old data with the new object
Stimulus	The user has queried for a Teacher Assistant to be deleted with new valid attribute data
Response	If the Teacher Assistant is not modified the method returns false. If the Teacher Assistant is modified the method returns true
Comments	Because the Teacher Assistant must already be displayed for the user to have the option to the modify, the method does not need to check if they exist

Test cases

Admin Test cases

Test One: Professor Manager - Add

Input:

- Add a new professor with the requested information

Tests:

1. Test for inputs with correct information in the correct format
2. Test for inputs with correct information in the incorrect format
3. Test for inputs with incorrect information in the correct format

Output:

OK or error message indicating an error with information

Test Two: Professor Manager - Modify

Input:

- Modify an existing professor's information

Tests:

1. Test modify by changing information to different info with correct format
2. Test modify by changing information to different info with incorrect format
3. Test modify by changing information to different info with incorrect info

Output:

OK or error message indicating an error with modifying information

Test Three: Professor Manager - Delete

Input:

- Delete a professor from the database

Tests:

1. Delete a professor

Output:

- OK or error message indicating failed attempt to delete

Test Four: Staff Manager - Add**Input:**

- Add a new Staff with the requested information

Tests:

1. Test for inputs with correct information in the correct format
2. Test for inputs with correct information in the incorrect format
3. Test for inputs with incorrect information in the correct format

Output:

- OK or error message indicating an error with information

Test Five: Staff Manager - Modify**Input:**

- Modify an existing Staff's information

Tests:

1. Test modify by changing information to different info with correct format
2. Test modify by changing information to different info with incorrect format
2. Test modify by changing information to different info with incorrect info

Output:

- OK or error message indicating an error with modifying information

Test Six: Staff Manager - Delete**Input:**

- Delete a Staff from the database

Tests:

1. Delete a Staff

Output:

- OK or error message indicating failed attempt to delete

Test Seven: Teacher Assistant Manager - Add**Input:**

- Add a new Teacher Assistant with the requested information

Tests:

1. Test for inputs with correct information in the correct format
2. Test for inputs with correct information in the incorrect format
3. Test for inputs with incorrect information in the correct format

Output:

- OK or error message indicating an error with information

Test Eight: Teacher Assistant Manager - Modify**Input:**

- Modify an existing Teacher Assistant's information

Tests:

1. Test modify by changing information to different info with correct format
2. Test modify by changing information to different info with incorrect format
3. Test modify by changing information to different info with incorrect info

Output:

- OK or error message indicating an error with modifying information

Test Nine: Teacher Assistant Manager - Delete**Input:**

- Delete a Teacher Assistant from the database

Tests:

1. Delete a Teacher Assistant

Output:

- OK or error message indicating failed attempt to delete

*Professor/TA Test Case***Test Ten: Add Student Grades****Input:**

- Add grades for student into system

Tests:

1. Test for adding grades with correct format and information
2. Test for adding grades with correct format and improper information
3. Test for adding grades with improper format and correct information

Output:

- Ok or error message indication fail information being inputted

Test Eleven: Modify Student Grades**Input:**

- Add grades for student into system

Tests:

1. Test for modifying grades with correct format and information
2. Test for modifying grades with correct format and improper information
3. Test for modifying grades with improper format and correct information

Output:

- Ok or error message indication fail information being inputted

Test Twelve: Delete Student Grades**Input:**

- Add grades for student into system

Tests:

1. Test for deleting grades with correct format and information
2. Test for deleting grades with correct format and improper information
3. Test for deleting grades with improper format and correct information

Output:

- Ok or error message indication fail information being inputted

About the Dev Team



señor Beast
its what we do

AUSTIN WALTER

Dev crew team lead, former UofSC-Aiken Help Desk extraordinaire, web design enthusiast. Check out his current side-project at <https://awalter7.github.io/main-blog/public/index.html>



Josh
Looks good to me

JOSH HOLM

Savannah River Site code-team member with a super sweet DoE clearance. Doesn't believe in laptops. Loves angsty-rock music and Asian food.



989onan
niceeee

ONAN CHEW

Java junkie. Makes Minecraft-mods 24/7 (has hundreds-of-thousands of downloads). Hopes to one day upload his consciousness and live inside Project Ozone 3. Check out his work at <https://www.curseforge.com/members/989onan/projects>



adam123
submitted.... g'nite nerds

ADAM BOESE

Works at UofSC-Aiken VMSS as auditor/certifier. Writes really long super boring process manuals (like this one) all day. Grades math homework and easily loses patience you darn whippersnappers. Network+, Security+, GIAC GFACT certified.

NOTES:

This page intentionally left blank
