**Project Design**

**CMSC 495 #6380**

**9/13/20**

**Group #5**

**Members:**

**David Roberts**

**Tiago Feitosa**

**Nicholas Crown**

**Contents**

[**Revision History** 3](#_Toc51095640)

[**Project Design for Martial Arts Attendance Tracking Application** 5](#_Toc51095641)

[**Start-Up Event-Tracing** 5](#_Toc51095642)

[***Scenario: Startup*** 5](#_Toc51095643)

[**Shut-Down Event-Tracing** 5](#_Toc51095644)

[***Scenario: Shutdown*** 5](#_Toc51095645)

[**Normal Operation Event-Tracing** 6](#_Toc51095646)

[***Scenario 1: Correct Authentication*** 6](#_Toc51095647)

[***Scenario 2: New Account Record Creation - Student*** 6](#_Toc51095648)

[***Scenario 3: New Account Record Creation - Instructor*** 7](#_Toc51095649)

[***Scenario 4: Loading Existing Account Record*** 7](#_Toc51095650)

[***Scenario 5: Successful Modification of Existing Account Record*** 8](#_Toc51095651)

[***Scenario 6: Deletion of Existing Account Record*** 8](#_Toc51095652)

[***Scenario 7: New Class Creation*** 9](#_Toc51095653)

[***Scenario 8: Loading Existing Class Record*** 10](#_Toc51095654)

[***Scenario 9: Successful Modification of Existing Class Record*** 10](#_Toc51095655)

[***Scenario 10: Deletion of Existing Class Record*** 11](#_Toc51095656)

[***Scenario 11: Loading Class Attendance*** 11](#_Toc51095657)

[***Scenario 12: Modifying Class Attendance*** 12](#_Toc51095658)

[***Scenario 13: Calendar Menu Selection*** 12](#_Toc51095659)

[**Error-Handling Event-Tracing** 14](#_Toc51095660)

[***Scenario 1: Incorrect Authentication*** 14](#_Toc51095661)

[***Scenario 2: Insufficient Information for New Account Record Creation*** 14](#_Toc51095662)

[***Scenario 3: Duplicate Account Creation Handling*** 14](#_Toc51095663)

[***Scenario 4: Failure to Modify Existing Account Record*** 15](#_Toc51095664)

[***Scenario 5: Insufficient Information for New Class Creation*** 15](#_Toc51095665)

[***Scenario 6: Duplicate Class Creation Handling*** 16](#_Toc51095666)

[***Scenario 7: Failure to Modify Existing Class Record*** 16](#_Toc51095667)

[**Class Design (Informal Pseudocode)** 18](#_Toc51095668)

[***Login Subsystem*** 18](#_Toc51095669)

[***Classes Subsystem*** 18](#_Toc51095670)

[***Account Subsystem*** 19](#_Toc51095671)

[***Attendance Subsystem*** 21](#_Toc51095672)

[***Validation Subsystem*** 22](#_Toc51095673)

[**Unresolved risk and mitigation** 23](#_Toc51095674)

[***Spoofing*** 23](#_Toc51095675)

[***Tampering*** 23](#_Toc51095676)

[***Repudiation*** 23](#_Toc51095677)

[***Information Disclosure*** 24](#_Toc51095678)

[***Denial of service*** 24](#_Toc51095679)

[***Elevation of privilege*** 24](#_Toc51095680)

# **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Revisor** | **Changes made** |
| 1.0 | 9/13/20 | Nicholas Crown | Proposed event-tracing scenarios |
| 1.0.2 | 9/15/20 | Nicholas Crown | Reviewed and simplified scenarios and subsystems |
| 1.1 | 9/13/20 | Tiago Feitosa | Added 3 new scenarios  Added Class Designs |
| 1.1.2 | 9/15/20 | Tiago Feitosa | Reviewed Class Designs according to group decisions.  Added startup and shutdown scenarios |
| 1.2 | 09/15/20 | David Roberts | Added Risks and Mitigation |
| 1.3 | 09/15/20 | Nicholas Crown | Modified scenarios for simplified design and performed document cleanup |
| 1.3.2 | 9/15/20 | Nicholas Crown | Added drawings for normal scenarios 1-5 and error handling 1-4 |
| 1.3.3 | 9/15/20 | Nicholas Crown | Reorganized scenarios into startup, shutdown, normal operation, and error handling per rubric. Added remaining scenario diagrams and performed document cleanup |
| 1.4 | 9/15/20 | Tiago Feitosa | Updated subsystem pseudocode to include Validation subsystem |

# **Project Design for Martial Arts Attendance Tracking Application**

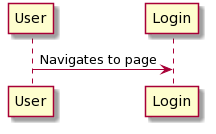
## **Start-Up Event-Tracing**

### ***Scenario: Startup***

Description: The user navigates to the index url and enters the login page.

Precondition: Web servers and database servers must be up and running.

Post-Condition: The application initializes a session and connects to the database.



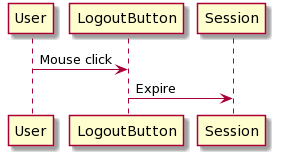
## **Shut-Down Event-Tracing**

### ***Scenario: Shutdown***

Description: The user logs out of the web application.

Precondition: The user has navigated to the web application location and logged in with an account. A session is currently active.

Post-Condition: The web application ends a session.



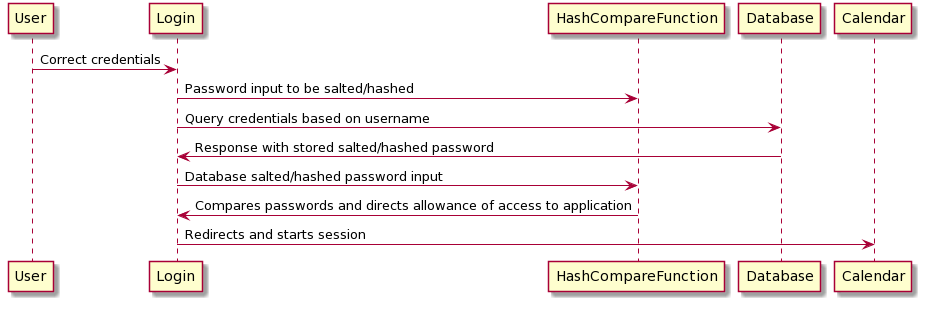
## **Normal Operation Event-Tracing**

### ***Scenario 1: Correct Authentication***

Description: The user enters the correct username and password of an account.

Precondition: The user has navigated to the web application location in their browser and the account exists in the database.

Post-Condition: The web application performs credential comparison and permits access to the user, loading the class calendar schedule view and allowing access to the application.

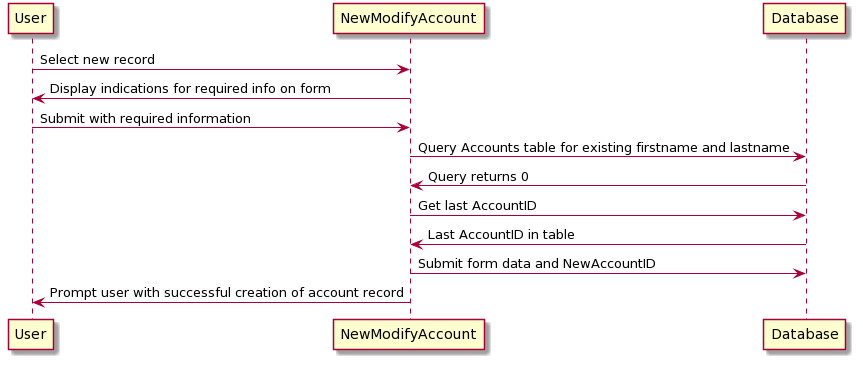


### ***Scenario 2: New Account Record Creation - Student***

Description: The user navigates to the New/Modify Account menu selection, selects new account record, and submits a form with all require information specified.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application accepts the form data, creates a new record in the Accounts table of the database with the Boolean statement false for the instructor field, and displays a message to the user notifying them that the operation was successful.

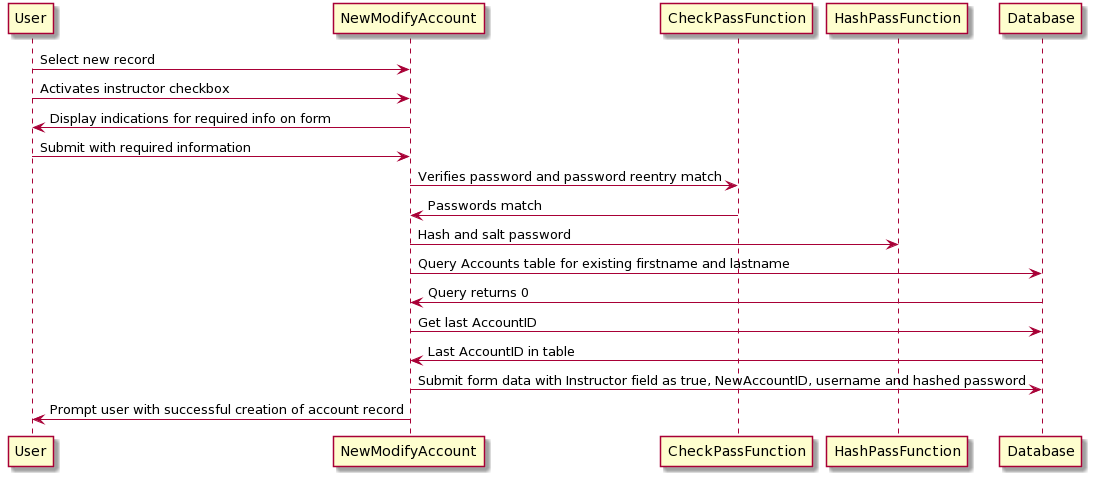


### ***Scenario 3: New Account Record Creation - Instructor***

Description: The user navigates to the New/Modify Account menu selection, selects new account record, and submits a form with all the required information specified and the instructor checkbox selected.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application accepts the form data, creates a new record in the Accounts table of the database with the Boolean statement as true for the instructor field, stores supplied password credentials as a salted hash in the database, and displays a message to the user notifying them that the operation was successful.

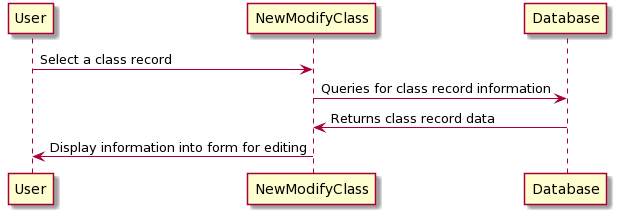


### ***Scenario 4: Loading Existing Account Record***

Description: The user navigates to the New/Modify Account menu selection and selects an existing account record from the search/drop-down box.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application queries the database and populates the information into the form fields.

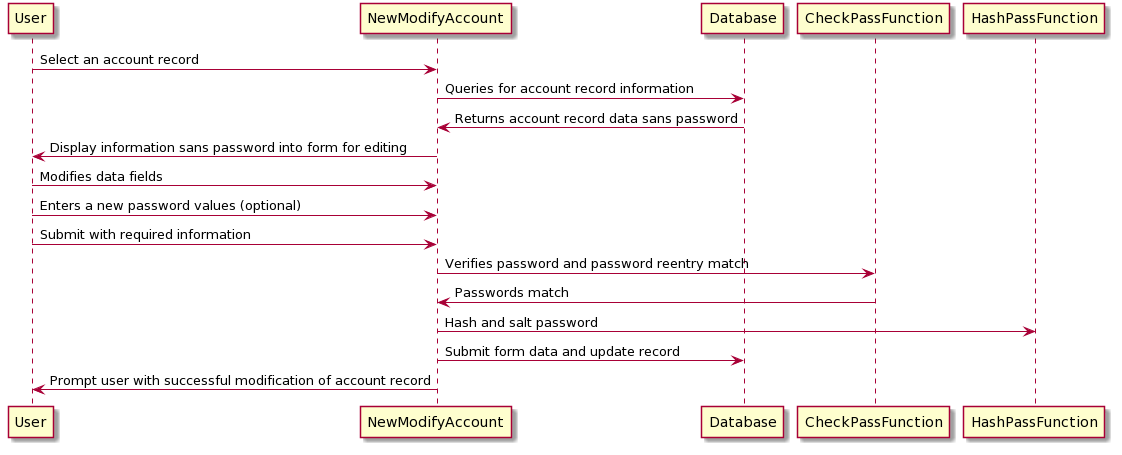


### ***Scenario 5: Successful Modification of Existing Account Record***

Description: The user changes information populated on the form, but required information is still present.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Account menu selection and selected an existing account record.

Post-Condition: The web application accepts the form data and makes changes to the existing record.

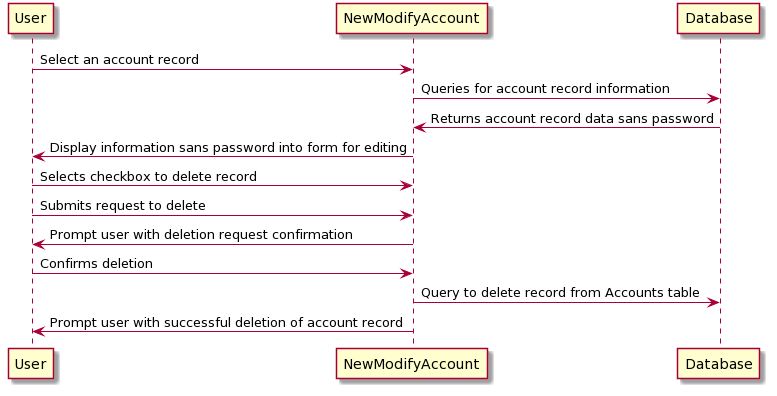


### ***Scenario 6: Deletion of Existing Account Record***

Description: The user selects the delete checkbox on the existing record.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Account menu selection and selected an existing account record.

Post-Condition: The web application confirms deletion of record by prompting the user response to confirm the action, then sends a query to delete the specified record from the database.

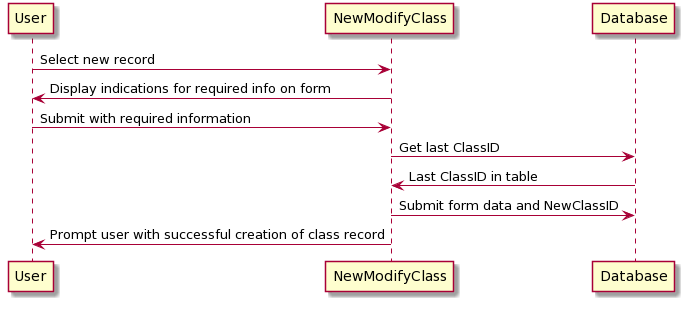


### ***Scenario 7: New Class Creation***

Description: The user navigates to the New/Modify Class menu selection, selects new class, and submits a completed form.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application accepts the form data, creates a new record in the Classes table of the database, and displays a message to the user notifying them that the operation was successful.

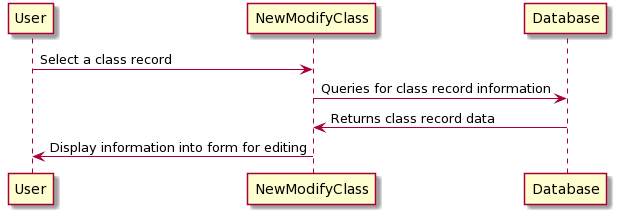


### ***Scenario 8: Loading Existing Class Record***

Description: The user navigates to the New/Modify Class menu selection and selects an existing account record from the search/drop-down box.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application queries the database and populates the information into the form fields.

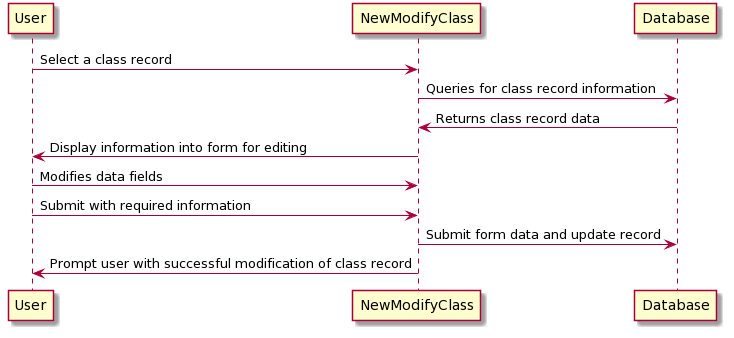


### ***Scenario 9: Successful Modification of Existing Class Record***

Description: The user changes information populated on the form, but required information is still present.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Class menu selection and selected an existing account record.

Post-Condition: The web application accepts the form data and makes changes to the existing record.

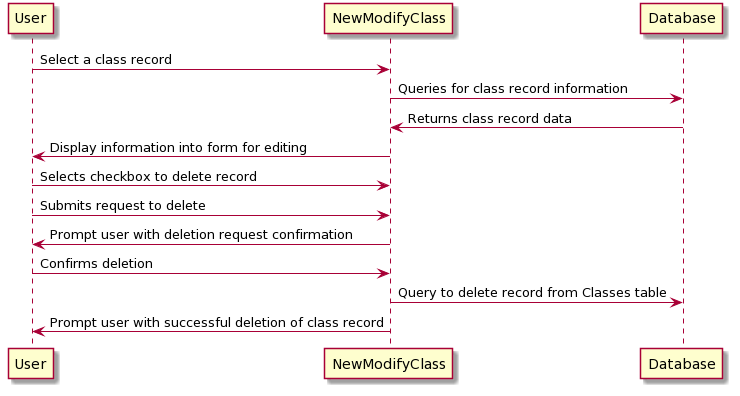


### ***Scenario 10: Deletion of Existing Class Record***

Description: The user selects the delete checkbox on the existing record.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Class menu selection and selected an existing account record.

Post-Condition: The web application confirms deletion of record by prompting the user response to confirm the action, then sends a query to delete the specified record from the database.

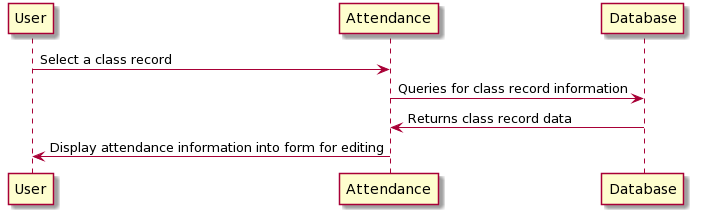


### ***Scenario 11: Loading Class Attendance***

Description: The user navigates to the Class Attendance menu selection and selects a class.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application queries the Classes table for the current class record and displays current attendance record to the side of the attendance form. The search/drop-down box highlights current attendees and instructors.

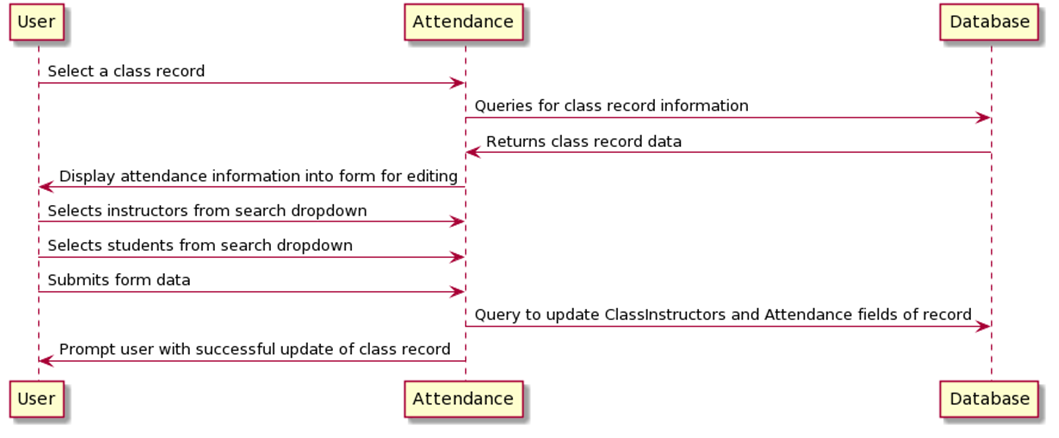


### ***Scenario 12: Modifying Class Attendance***

Description: The user selects/deselects instructors and students from the search/drop-down boxes and submits form.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the Class Attendance menu selection, and selected a class.

Post-Condition: The web application updates the ClassInstructors and Attendance fields of the selected class record in the Classes table of the database with the selected AccountIDs.

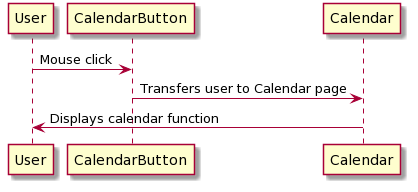


### ***Scenario 13: Calendar Menu Selection***

Description: The user clicks on the Calendar button on any page of the system

Precondition: The user has navigated to the web application location and logged in.

Post-Condition: The web application shows the calendar page.



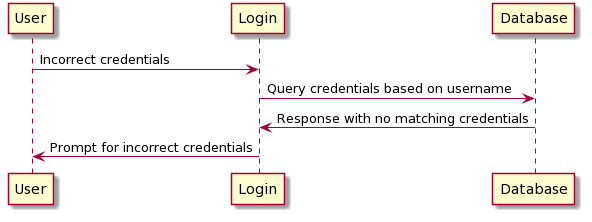
## **Error-Handling Event-Tracing**

### ***Scenario 1: Incorrect Authentication***

Description: The user enters an incorrect username or password.

Precondition: The user has navigated to the web application location in their browser.

Post-Condition: The web application denies access to the user, stating that they have entered an incorrect username or password.

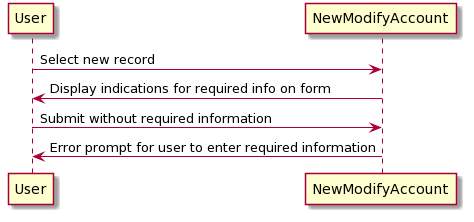


### ***Scenario 2: Insufficient Information for New Account Record Creation***

Description: The user navigates to the New/Modify Account menu selection, selects new account record, and attempts to submit form without required information.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application prevents the user from submitting the form and displays an error notifying the user that the form is incomplete.

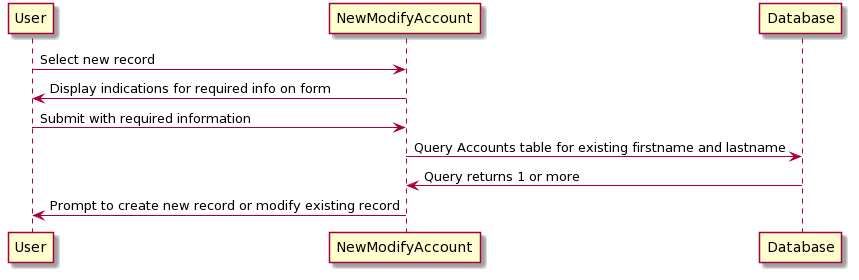


### ***Scenario 3: Duplicate Account Creation Handling***

Description: The user attempts to submit a form with a first name and last name combination that already exists.

Precondition: The user has navigated to the web application location, logged in with an account, and navigated to the New/Modify Account menu selection.

Post-Condition: The web application requests the user to confirm creating another account entry in the database or updating the existing one.

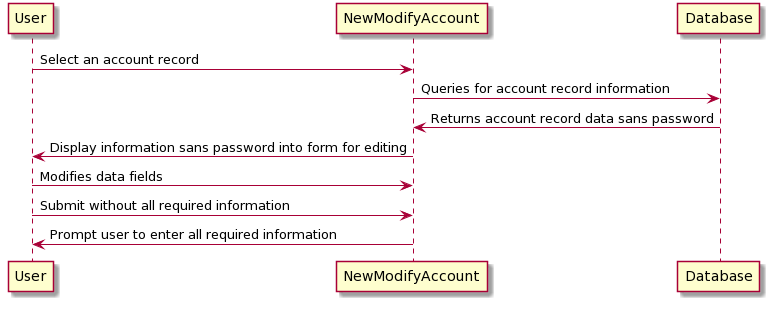


### ***Scenario 4: Failure to Modify Existing Account Record***

Description: The user changes information populated on the form but has removed required information.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Account menu selection and selected an existing account record.

Post-Condition: The web application prevents the user from submitting the form and displays an error notifying the user that the form is incomplete.

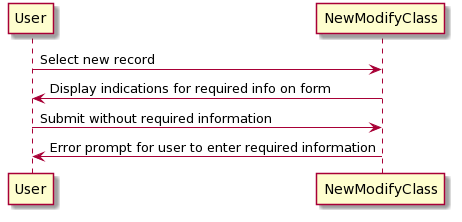


### ***Scenario 5: Insufficient Information for New Class Creation***

Description: The user navigates to the New/Modify Class menu selection, selects new class, and attempts to submit an incomplete form.

Precondition: The user has navigated to the web application location and logged in with an account.

Post-Condition: The web application prevents the user from submitting the form and displays an error notifying the user that the form is incomplete.

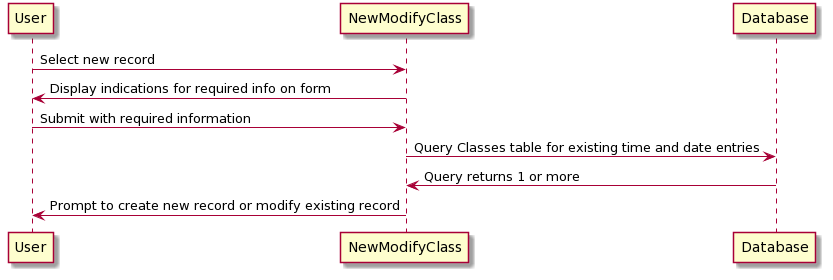


### ***Scenario 6: Duplicate Class Creation Handling***

Description: The user attempts to submit a form with the same time/date slot as another class

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Class menu selection, and selected to create a new class.

Post-Condition: The web application requests the user to confirm creating another class entry in the database or to update the existing one.

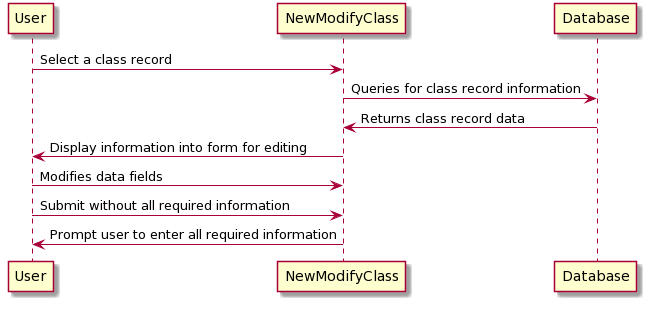


### ***Scenario 7: Failure to Modify Existing Class Record***

Description: The user changes information populated on the form but has removed required information.

Precondition: The user has navigated to the web application location, logged in with an account, navigated to the New/Modify Class menu selection and selected an existing account record.

Post-Condition: The web application prevents the user from submitting the form and displays an error notifying the user that the form is incomplete.



## **Class Design (Informal Pseudocode)**

### ***Login Subsystem***

Class Login {

Void CreateUser (user, password) {

User = user;

Password = password;

ID = GetDatabaseNewID;

DBquery.create\_new\_user(ID, user, password);

}

Boolean AuthenticateUser (user, password) {

DBquery.parameterize(user, password); // Parameterization to avoid DB attacks

If DBquery.user.Password == password

Return True

}

Void initializeApp () {

CreateDBConnection();

Date today = Date.today;

Calendar calendar(today);

Call calendar.GetTotalActiveStudents(today.month);

Call calendar.GetNewStudentsMonth(today.month);

Call calendar.GetDropStudentsMonth(today.month);

Call calendar.GetActiveClasses(today);

}

### ***Classes Subsystem***

Class Classes {

Int ID; {get/set}

String Name; {get/set}

Date StartDate; {get/set}

Date EndDate; {get/set}

Weekday [ ] WeekDays; {get/set}

Time StatTime; {get/set}

Time EndTime; {get/set}

Boolean Status; True=Open/False=Closed {get }

Boolean ValidateString(string) {

DBquery.validate\_string\_before\_submitting\_to\_DB;

}

Void CreateClass (name, startdate, enddate, starttime, endtime, weedays) {

ValidateString(name);

ID = getDatabaseNewID;

Name = name;

StartDate = startdate;

DndDate = enddate;

StartTime = starttime;

}

Void UpdateStatus (date) {

If date > EndDate

Status = False;

}

Void Delete(Classes) {

DBquery.delete\_ Classes (Classes);

}

Void Update(Classes) {

DBquery.update\_ Classes (Classes);

}

Student Get(Classes) {

Return DBquery.get\_Classes (Classes);

}

}

### ***Account Subsystem***

Enum Rank {

WB,

1S,

2S,

3S,

4S,

5S,

6S,

1D,

2D,

3D,

4D,

5D,

6D,

7D,

8D,

9D,

10D

}

Class Account {

Int ID; {get}

Boolean instructor; {get/set}

String FirstName; {get/set}

String LastName; {get/set}

String ParentalContact; {get/set}

Date DateOfBirth; {get/set}

Int Age; {get}

String PhoneNumber; {get/set}

String Address; {get/set}

String Email; {get/set}

Rank Belt; {get/set}

Boolean Status; {get/set}

String Notes; {get/set}

Attendance attendance;

Boolean ValidateString(string) {

DBquery.validate\_string\_before\_submitting\_to\_DB;

}

Constructor() {}

Void Add(student) {

DBquery.add\_student(student);

}

Void Delete(student) {

DBquery.delete\_student(student);

}

Void Update(student) {

DBquery.update\_student(student);

}

Student Get(student) {

Return DBquery.get\_student(student);

}

}

***Calendar Subsystem***

Class Calendar {

Int CurrentDay; {get/set}

Classes [ ] ActiveClasses;

Constructor(date) {}

Int GetTotalActiveStudents (date) {

DBquery.get\_number\_students;

Return TotalActiveStudents;

}

Int GetNewStudentsMonthly(date) {

DBquery.get\_month\_new\_students(date);

}

Int GetDropsMonthly(date) {

DBquery.get\_month\_drop\_students(date);

}

Classes [] GetActiveClasses(date) {

Return ActiveClasses;

}

}

### ***Attendance Subsystem***

Class Attendance {

Map <Date/Boolean> attendance;

Void AddAttendanceToStudent(student/instructor, date, boolean) {

Student.attendance.add(date, boolean);

}

Void ResetAttendanceToStudent(student/instructor) {

Student.Attendance.clear();

}

}

### ***Validation Subsystem***

Class Validation {

Void HashCompare(password) {

If password == DBquery.user.password

Return true;

}

Void CheckPass(password) {

If password == DBquery.user.password

Return true;

}

**}**

## **Unresolved risk and mitigation**

### ***Spoofing***

**Unresolved risk.**

The user is authenticated using a username and strong password. There are currently no measures in place to prevent common password attacks such as a brute force attack.

**Mitigation.**

Add a maximum number of login attempts before an account becomes locked. Two factor authentication would also help mitigate this risk.

### ***Tampering***

**Unresolved risk.**

An instructor can demote all other instructors to students, giving them sole access to the system.

**Mitigation.**

Reauthentication should occur when demoting or removing an instructor. An instructor should only be allowed to modify the credentials of an instructor of the same rank or lower.

### ***Repudiation***

**Unresolved risk.**

The system is vulnerable to replay attacks during the login process.

**Mitigation**.

Using a nonce when hashing the password will mitigate this risk.

### ***Information Disclosure***

**Unresolved risk.**

Any instructor can view the personal information of all other instructors. This creates a risk that a lower belt instructor can view or modify the information of a higher belt instructor.

**Mitigation.**

Instructors should only be able to view the information students and instructors of their rank or lower.

### ***Denial of service***

**Unresolved risk.**

The application is currently at risk of a DDOS attack.

**Mitigation.**

The use of a third-party DDOS protection service would best mitigate this risk.

### ***Elevation of privilege***

**Unresolved risk.**

There are no checks done when promoting a student to an instructor. A student could promote themselves to an instructor position if they gained access to the system.

**Mitigation.**

The user should be reauthenticated when promoting an instructor or promoting a student to an instructor.