Software Requirements Specification

for

CoachConnect

**Version 1.0 approved**

**Prepared by Pao Xiong, Adam Smith, Brian Lueskow, Tim Durkee**

**PABT, Inc**

**05/09/2017**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1.** **Introduction 1**

1.1 Purpose 1

1.2 References 1

**2.** **Overall Description 1**

2.1 User Classes and Characteristics 1

2.2 Operating Environment 1

2.3 Design and Implementation Constraints 2

2.4 Assumptions and Dependencies 2

**3.** **External Interface Requirements 2**

3.1 User Interfaces 2

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4.** **System Use Cases 3**

4.1 Use case name and identifier 4

4.2 Making Schedule with a Tutor (U5) 4

4.3 Deposit money into ATM (U3) 5

**5.** **Other Nonfunctional Requirements 5**

5.1 Performance Requirements 5

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 6

**6.** **Other Requirements 6**

**7.** **System Requirements Chart 6**

**Appendix A: Analysis Models 6**

**Appendix B: To Be Determined List 6**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Adam Smith | 02/26/17 | Adding initial ideas and requirements to Sections 2,3, and 5 | 0.1 |
| Pao Xiong | 02/26/17 | Add Use Case ‘U2’; 4.2 | 0.1 |
| Brian Lueskow | 02/26/17 | Add Introduction/Purpose 1.1 | 0.1 |
| Tim Durkee | 02/26/17 | Add External Interface Requirements Section 3 | 0.1 |
| Tim Durkee | 04/02/17 | Add Use Case ‘U3’ & ‘U4’; 4.3, 4.4 | 0.2 |
| Tim Durkee | 04/03/17 | Add Use Case ‘U5’; 4.5 | 0.3 |
| Tim Durkee | 04/08/17 | Add Use Case ‘U6’ & ‘U7’;4.6,4.7 | 0.4 |
| Brian Lueskow | 05/07/17 | Add Use Case ‘U8’ | 0.4 |
| Tim Durkee | 04/30/17 | Add Use Case ‘U9’,’U10’,’U11’; 4.8,4.9,4.11 | 0.5 |
| Tim Durkee | 05/08/17 | Add Use Case ‘U12’; 4.12 | 0.6 |
| Tim Durkee | 05/08/17 | Update *External User Interface* | 0.6 |
| Tim Durkee | 05/08/17 | Update *Other Nonfunctional Requirements* | 0.6 |
| Tim Durkee | 05/08/17 | Update *System Requirements Chart* | 0.6 |
| Tim Durkee | 05/08/17 | Update Appendices | 0.6 |
| Tim Durkee | 05/09/17 | Add Use Case ‘U13’ | 0.7 |

**Introduction**

## Purpose

## To Identify the product in which will be used for this project is Visual Studio 2015 version and a form of database together, to better accomplish the project in the best way for what the design is intended to perform within the project. To best Describe the purpose of this program is to provide a faster more subject focused way for students to be able to schedule a time with an academic coach for specific subject. This can also allow students to better manage their time for homework.

## This will also provide staff to schedule hours for the lab more accurately. The software being used to design the project is the language C# and a form a Database Development to hold the records and information for the students and academic coaches.

## References

None

# Overall Description

## User Classes and Characteristics

**Student :**

Students can add current classes and further drill down the filters to find an appropriate tutor for the level of the class. Technical expertise for this user will be minimal since they will be the heaviest users of the program and will be the driving force behind the product. Students security level will be minimal and will be locked down the most

**Coach:**

Coaches will have similar security and expertise level needed as students. The will be able to add their certifications and rank their abilities from strongest to weakest. Requirements of these users are experience and/or educational background. These users will also be “verified” users once they pass their background check.

**Administrator**

Administrators are involved with their overall management of the systems and payments. They will be hired by the company to oversee everything while working with Student and Teacher users to address issues that arise

## Operating Environment

The application requires a minimum of Windows 7 using the .NET framework version 4 or higher. We recommend a 1 GHz or faster processor, 2 GB of RAM and 1 GB of available hard disk space.

## Design and Implementation Constraints

Valid internet connection necessary to use the software. Without constant internet connection, the database will not be available on any device and application will not function as desired.

Database will not have data from any other schools. It’s only limited for Northeast Wisconsin Technical College

## Assumptions and Dependencies

Software is intended to be used only on a Windows Based machine but future versions will be compatible amongst other platforms. It could possibly be used in the future on mobile products but may not operate correctly.

Future implementation of ads and marketing may be added to supplement ongoing project funding.

# External Interface Requirements

## User Interfaces

* All forms will display tab control to allow users to move back and forth easily
* Font will be Microsoft Sans Serif, 8.25 size
* Sessions layout will be datagrids
* Editing will lock all other aspects of program until finished
* Default form color will be kept and not changed
* All monitor resolutions will be supported
* Closing form with ‘X’ will logout user. Exit the program from login screen or *exit* button
* User will need to validate profile images, application will not adjust for invalid URL.

## Hardware Interfaces

Application will only be working with monitor, mouse, and keyboard. Print options have been removed due to time constraints.

## Software Interfaces

At startup, the software will establish connect to a SQL Server database to access tutor and student records. As new users are created, their data will be saved to the database for future use. Schedules will also be created and maintained through this database.

## Communications Interfaces

Communication interfaces have not been implemented in this project

# System Use Cases

## 4.2 Making Schedule with a Tutor (U2 Pao)

1. **U2**
2. **Objective** – The student is making schedule to see his/her tutor by using the software. He/she will check for the appropriate time to choose and the system will save his/her schedule and send a notification message to the tutor’s email.
3. **Priority** – Low
4. **Source** – Mr. A (Student Supported Center)
5. **Actors** – Students, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. Student logs into the system
      2. Student clicks to make schedule
      3. System Checks for the available date & time
      4. Student selects the appropriate date/time to make schedule
      5. Student submits his/her schedule
      6. System blocks off the date and time so it will not get double booked
      7. System sends confirmation email to student and tutor
   2. **Alternative Flow 1** – At step 6.1.5 Student did not choose time
      1. Error message displayed for student to choose proper time
      2. Return to step 6.1.4
   3. **Alternative Flow 2** – At any step the user presses “cancel”
      1. System returns to the main menu
   4. **Alternative Flow 3** - At step 6.1.6 System encounters failure
      1. An error message is shown to the student
      2. Return to step 6.2
   5. **Exception Flow 1** –
      1. Database is locked due to backup in progress.
7. **Includes**
   1. none
8. **Preconditions** – Student is logged in
9. **Post conditions** – the successful message is displayed to the student and the schedule is made
10. **Notes/Issues** – None

## 4.3 Making Appointment with a Coach (U3 Adam)

1. **U3**
2. **Objective** – The student is making an appointment to see his/her coach by using the software. He/she will select for tap options to find an interesting coach such as Find Coach by Name, Find Coach by Interest, and Find Coach by Time.
3. **Priority** – Low
4. **Source** – Student Support Center
5. **Actors** – Students, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. The student logins to the system
      2. The system guides the student to the main page
      3. The student chooses an option tap to begin his/her making appointment
         1. Find Coach by Name
            1. If The student chooses the Find Coach by Name – He/she chooses a coach name from the dropdown list then the coach information is going to show up below.
            2. The system checks and retrieves the available date time.
            3. The student chooses for the appropriate date time to make an appointment.
            4. The student input a purpose message.
            5. The student submits his/her appointment by clicking on the Booking button.
         2. Find Coach by Interest
            1. If the student chooses the Find Coach by Interest – He/she chooses one of the coaches appeared.
            2. The system is going to retrieves that coach’ information and available time to a page
            3. The student chooses for the appropriate date time to make an appointment.
            4. The student input a purpose message
         3. Find Coach by Time
            1. If the student chooses the Find Coach by Time – He/she check on a day of week and a part of day.
            2. The system is going to retrieves that coach’ information.
            3. The student input a purpose message.
            4. The student submits his/her appointment by clicking on the Booking button.
      4. The system asks the School computer to save the schedule
      5. The system sends a notification email to the tutor’s email
   2. **Alternative Flow 1** - At step 6.1.3.1.5, 6.1.3.2.5, and 6.1.3.3.1 The student did not choose one or any date time or missing one or some inputs
      1. An error message is displayed telling the student to correct with the error
      2. The system return to step 6.1.3
   3. **Alternative Flow 2** – At any time the student click on the cancel
      1. The system returns to the main menu
   4. **Alternative Flow 3** – At step 6.1.6 the system encounters failure
      1. An error message is displayed to the student
      2. Return to step 6.1.2
   5. **Exception Flow 1**
      1. Database is locked according to backup in progress
7. **Includes**
   1. none
8. **Preconditions** – Student is logged in
9. **Post conditions** – the successful message is displayed to the student and the schedule is made
10. **Notes/Issues** – None

## 4.4Cancel an Appointment with a Coach(U4Pao)

1. **U4**
2. **Objective** – The student can make decision to cancel his/her appointment which is made with a coach any time after he/she has made.
3. **Priority** – Low
4. **Source** – Mr. A (Student Supported Center)
5. **Actors** – Students, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. Student logs into the system
      2. The system guides the student to the main page
      3. The system retrieves student’s information with the all appointments which are made
      4. The student chooses an appointment to cancel
      5. The student clicks on Cancel button
      6. The system shows a confirm box to confirm the cancellation
      7. The system shows a message box to let the student knows that the cancelation is made completely.
   2. **Alternative Flow 1** – At step 6.1.4 the student didn’t choose any appointment
      1. The system displays an error message box to let the student know.
      2. Return to step 6.1.2
   3. **Alternative Flow 2** – At step 6.1.6 the system encounters failure
      1. An error message is displayed to the student
      2. Return to step 6.1.2
   4. **Alternative Flow 3** - At step 6.1.6 System encounters failure
      1. An error message is shown to the student
      2. Return to step 6.2
   5. **Exception Flow 1** –
      1. Database is locked due to backup in progress.
7. **Includes**
   1. none
8. **Preconditions** – Student is logged in
9. **Post conditions** – the successful message is displayed to the student and the schedule is made
10. **Notes/Issues** – None

## 4.5 Create Student Schedule (U5)

1. **U5**
2. **Objective** – Display student schedule
3. **Priority** – High
4. **Source** – Student
5. **Actors** – Students, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. Student logs into the system
      2. Student clicks to view schedule
7. **Includes**
   1. none
8. **Preconditions** – Student is logged in
9. **Post conditions** – the successful message is displayed to the student and the schedule is made
10. **Notes/Issues** – None

## 4.6 Coach View Current Schedule (U6 - Tim)

1. **U6**
2. **Objective** – Display Coach Schedule
3. **Priority** – High
4. **Source** – Coach
5. **Actors** – Coach, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. Coach logs into the system
      2. Coach clicks tab to view schedule
      3. Coach’s tab displays current schedule
7. **Includes**
   1. none
8. **Preconditions** – Coach is logged in
9. **Postconditions** – Coach schedule will be displayed but not editable for the user
10. **Notes/Issues** – May need to add a function to print the schedule for the coach

## 4.7 Coach Update Availability (U7 - Tim)

1. **U7**
2. **Objective** – Update Availability Tab
3. **Priority** – High
4. **Source** – Coach
5. **Actors** – Coach, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. Coach Clicks Availability Tab
         1. Enable current selected options for availability.
      2. Coach updates day part
      3. Coach selects submit
      4. System updates filed in database
      5. Coach receives email confirming update
   2. **Alternative Flow** - At step 6.1.3, Coach does not update schedule
      1. Popup message, “No changes Made”
         1. Continue Flow
7. **Includes**
   1. none
8. **Preconditions** – Coach is logged in
9. **Postconditions** – Return to Home screen for the Coach
10. **Notes/Issues** –None

## 4.8 Create User Admin Tools (U8- Brian)

1. **U8**
2. **Objective** – To establish the create user within form that could multi-function so changes can be made within the database for new students and coaches information such as first name, middle name, and last name etc. Schedules for coaches will be accessible through this form as well and all information that is changed within the form can all be updated within the database with the proper permissions that will allow the changes to made and functions within the database.
3. **Priority** – High
4. **Source** – Requesting Student, Coach
5. **Actors** –Administrator
6. **Flow of Events** 
   1. **Basic Flow** The administrator would be able to make changes to the schedules, Students, courses etc.
      1. Coaches should be able to navigate through out the admin form freely with proper permissions to apply the proper changes for passwords and personal information for adding, updating or making student not available.
      2. Students should be limited but available with proper permissions to update their personal information in the system.
   2. **Alternative Flow** - -The administrator would have direct access to the database.
      1. -- pop up message “changes were made”
7. **Includes**
   1. --The form will include Buttons, Checkboxes, and text boxes for the bulk of the form functionality
   2. --Buttons will be for the adding users, to submit changes to the database, such as adding a user or changing personal information. The will be a logout to log off of the system to bring user back to log on screen and an Exit button that will completely log off of the system entirely.
   3. --Check boxes will be for applying Active, Admin, Student, or Coach. After these are applied to the database it then the user to access information dependant on what permissions are granted.
   4. --Text boxes will be for the add and update feature of the form and will be filled in according to what is required in the form. Changes can then be saved by hitting the submit button.
8. **Preconditions** – -Administrator must make sure proper changes were made to the database.
9. **Postconditions** – -Return to homepage to log in again or completely leave the system using buttons as a precondition to apply function.
10. **Notes/Issues** --Coaches and Student will always have different permissions unless that student is also a Academic Coach. Which in that case would be the only exception to the rule and must be cleared by proper management staff.

## 4.8 Update User Admin Tools (U8.2 - Brian)

## 4.9 Objective – To establish the update user within form that could multi-function so changes can be made within the database for new students and coaches information such as first name, middle name, and last name etc. Schedules for coaches will be accessible through this form as well and all information that is changed within the form can all be updated within the database with the proper permissions that will allow the changes to made and functions within the database.

1. **Priority** – High
2. **Source** – Requesting Student, Coach
3. **Actors** –Administrator
4. **Flow of Events** 
   1. **Basic Flow** The administrator would be able to make changes to the schedules, Students, courses etc.
      1. Coaches should be able to navigate through out the admin form freely with proper permissions to apply the proper changes for passwords and personal information for adding, updating or making student not available.
      2. Students should be limited but available with proper permissions to update their personal information in the system.
   2. **Alternative Flow** - -The administrator would have direct access to the database.
      1. -- pop up message “changes were made”
5. **Includes**
   1. --The form will include Buttons, Checkboxes, and text boxes for the bulk of the form functionality
   2. --Buttons will be for the adding users, to submit changes to the database, such as adding a user or changing personal information. The will be a logout to log off of the system to bring user back to log on screen and an Exit button that will completely log off of the system entirely.
   3. --Check boxes will be for applying Active, Admin, Student, or Coach. After these are applied to the database it then the user to access information dependant on what permissions are granted.
   4. --Text boxes will be for the add and update feature of the form and will be filled in according to what is required in the form. Changes can then be saved by hitting the submit button.
6. **Preconditions** – -Administrator must make sure proper changes were made to the database.
7. **Postconditions** – -Return to homepage to log in again or completely leave the system using buttons as a precondition to apply function.
8. **Notes/Issues** --Coaches and Student will always have different permissions unless that student is also a Academic Coach. Which in that case would be the only exception to the rule and must be cleared by proper management staff.

## 

## Update Profile Pic (U9 - Tim)

1. **U9**
2. **Objective** – Allow Coach to update their profile pic with a URL
3. **Priority** – Low
4. **Source** – Coach
5. **Actors** – Coach
6. **Flow of Events** 
   1. **Basic Flow**
      1. Coach clicks profile tab
      2. Click Edit Pic
         1. Disables all other tabs and group boxes until submit or cancel has been selected
         2. Enable Visibility Preview Button and Cancel Button
         3. Enable Textbox
      3. Paste or Type the URL link of Image
      4. User clicks Preview to verify if image is valid
         1. Disable visibility of Preview button
         2. Enable visibility of Submit button
         3. Update Picturebox of image from url
      5. Click Submit to update profile pic
         1. Updates database with new URL
         2. Disable textbox
         3. Disable visibility of Submit and Cancel buttons
         4. Enable Edit button
         5. Enable the other group boxes and tabs
   2. **Alternative Flow** - - at Step 6.1.4
      1. -- User selects cancel
         1. Disable textbox
         2. Disable visibility of Preview and Cancel
         3. Enable Edit button and group boxes
   3. **Alternative Flow** - - at Step 6.1.5
      1. User selects cancel before submitting
         1. Disable textbox
         2. Disable visibility of Preview and Cancel
         3. Enable Edit button and group boxes
7. **Includes**
   1. --none
8. **Preconditions** – - User has coach profile and is logged in as Coach
9. **Postconditions** – - Update the form with the new user information
10. **Notes/Issues** -- No validation check on the URL. It is the user’s discretion.

## 4.10 Edit Coach’s Personal Information(U10-Tim)

1. **U10**
2. **Objective** – Edit Coach’s Infor
3. **Priority** – Medium
4. **Source** – Coach
5. **Actors** – Coach
6. **Flow of Events** 
   1. **Basic Flow**
      1. User selects Profile tab
      2. User clicks Edit button in personal information group box
         1. Hide Edit button in group box
         2. Show Update and Cancel Button
         3. Enable Text boxes in the group box
         4. Disable all other group boxes
      3. User Edits the First, Middle and Last Name, Phone and Email
         1. When focus leaves textbox, string that is in said textbox is validated to ensure data integrity
      4. User selects Update
         1. Coach’s information is updated in database
         2. Hide Update and Cancel Buttons
         3. Enable Edit Button
         4. Enable all other group boxes
   2. **Alternative Flow** - - at Step 6.1.3
      1. User inputs invalid Characters in textbox
         1. Validation class will return a false value from appropriate method
         2. Error under invalid textbox will appear
         3. User can only change the text from the textbox or hit cancel
7. **Includes**
   1. --
8. **Preconditions** – -User is logged in
9. **Postconditions** – -Text fields are formatted for viewing
10. **Notes/Issues** --Updates the entire form of any personal changes

## 4.11 Update Password (U11-Tim)

1. **U11**
2. **Objective** – Update Password for user
3. **Priority** – High
4. **Source** – User
5. **Actors** – Coach
6. **Flow of Events** 
   1. **Basic Flow**
      1. User clicks Profile Tab
      2. user clicks Reset button on the Reset Password
         1. New form pops up. Uses *form.showdialog();* to force user to update or cancel and not select any other areas of the form.
      3. Input New password and confirmation
      4. Select Update
         1. Verify that salted hashes are a match
         2. Update database
         3. Close Window
   2. **Alternative Flow** - - at step 6.1.4
      1. -- User inputs mismatched passwords
         1. Popup notifying user
         2. Make user close popup
         3. Clear passwords in text boxes
   3. **Alternative Flow** - - at step 6.1.4
      1. User hits cancel
         1. Close form
7. **Includes**
   1. --
8. **Preconditions** – - User has coach profile and is logged in as Coach
9. **Postconditions** – - Password is encrypted and updated.
10. **Notes/Issues** --None

## 4.12 Edit Availability (U12-Tim)

1. **U12**
2. **Objective** – Update Availability for Coach
3. **Priority** – Medium
4. **Source** – User
5. **Actors** – Coach
6. **Flow of Events** 
   1. **Basic Flow**
      1. User clicks Availability Tab
      2. User clicks *Edit* 
         1. Checkboxes are then enabled
         2. Hide visibility of the *Edit* button. Show visibility of *None, All, Submit, Cancel*
      3. User checks/unchecks days for availability
      4. User clicks *Submit*
         1. Clear previous coaches availability in database
         2. Cycle through all checkboxes and update database accordingly
         3. Hide visibility of the *None, All, Submit, Cancel* and show visibility of the *Edit* button
   2. **Alternative Flow** - - at step 6.1.4
      1. -- User Clicks Cancel
         1. Reset checkboxes to original value when *Edit* was clicked.
   3. **Alternative Flow** - - at step 6.1.4
      1. User clicks *All*
         1. Check all checkboxes
   4. **Alternative Flow** - - at step 6.1.4
      1. User clicks *All*
         1. Uncheck all checkboxes
7. **Preconditions** – - User has coach profile and is logged in as Coach
8. **Postconditions** – - User Availability updated
9. **Notes/Issues** --None

## 4.13 View Coach - Sessions/rosters (U13-Tim)

1. **U13**
2. **Objective** – View Rosters and sessions
3. **Priority** – Low
4. **Source** – User
5. **Actors** – Coach
6. **Flow of Events** 
   1. **Basic Flow**
      1. User clicks Current Schedule Tab
      2. User double clicks session
         1. Dependent on Session clicked, will display a message box that will show current sessions assigned to coach
      3. User clicks ok to exit message box
      4. **Alternative Flow** - - at step 6.1.2.1
      5. -- If coach is assigned to session but no roster has been created, message will populate saying that no students have signed up yet for this session
7. **Preconditions** – - User has coach profile and is logged in as Coach
8. **Postconditions** – - User Availability updated
9. **Notes/Issues** --None

## 4.14 Session Admin form to update session details and roster (U14-Adam)

1. **U14**
2. **Objective** – View and update session details and rosters
3. **Priority** – Medium
4. **Source** – User
5. **Actors** – Admin
6. **Flow of Events** 
   1. **Opening form**
      1. User enters Admin section
      2. User clicks on Session tab
   2. **Edit existing session**
      1. If desired session is displayed, user double-clicks on session.
      2. User updates session details
      3. User adds new enrolled students (if desired)
      4. User removes non-enrolled students (if desired)
      5. User clicks OK to save changes
   3. **Create new session**
      1. User clicks New Session button
      2. User updates session details
      3. User adds new enrolled students
      4. User clicks OK to save changes
7. **Preconditions** – - User is designated as an Admin (under User Admin section).
8. **Postconditions** – - User Availability updated
9. **Notes/Issues** -- None

## 4.15 Session Admin form to update session details and roster (U15-Adam)

1. **U15**
2. **Objective** – User selects their role from a dropdown
3. **Priority** – High
4. **Source** – Dev Team
5. **Actors** – All users
6. **Flow of Events** 
   1. User selects the desired role from a combo box
   2. User clicks OK.
7. **Preconditions**
   1. User is registered in the database and is assigned to a minimum of one role (Student, Coach, or Admin).
   2. User is logged in with a valid username and password.
8. **Postconditions** – - The correct starting page will load based on the selected role.
9. **Notes/Issues** -- The role list will populate at runtime with the user’s assigned roles.

## 4.16 Session Admin form to update session details and roster (U16-Adam)

1. **U16**
2. **Objective** – User logs in to the .
3. **Priority** – High
4. **Source** – Dev Team
5. **Actors** – All users
6. **Flow of Events** 
   1. User enters his/her assigned username.
   2. User enters his/her password.
   3. User clicks OK.
   4. If the user enters incorrect credentials, the user tries again.
   5. If the user is flagged to “reset password”, user enters the desired new password in the next window.
7. **Preconditions**
   1. Program is opened normally and no load errors have occurred.
   2. User is registered in the database with a valid username and password.
8. **Postconditions**
   1. The entered password is encrypted using a salted hash and compared against the encrypted hash stored in the database.
   2. If the credentials are valid, the role page will load and display the user’s available options.
9. **Notes/Issues** -- None

# Other Nonfunctional Requirements

## Performance Requirements

See section 7 requirements 23-27

## Safety Requirements

Software should comply with all federal and state rules regarding educational and personal data

See section 7 requirements 25-32

## Security Requirements

(Add generic requirement for salted hash)

See section 7 requirements 55-62

## Software Quality Attributes

See section 7 requirements 55-62

# Other Requirements

None.

# System Requirements Chart

*< Include a* ***table*** *in this section with the following columns:*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *ID* | *Priority* | *Type* | *Source* | *Contained in Use Case* | *Description* |
| *DB1* | *High* | *F* | *Adam Smith (Dev Team)* | *U15* | *Create specific user types to limit access (Admin, Tutor, Student). Roles should be selected at login.* |
| *DB2* | *Med* | *F* | *Tim Durkee*  *(Dev Team)* | *U12* | *GUI layout for Coach Availability* |
| *DB3* | *High* | *F* | *Tim Durkee*  *(Dev Team)* | *U6* | *GUI layout for Coach Schedule* |
| *DB4* | *Low* | *F* | *Tim Durkee*  *(Dev Team)* | *U9* | *Update Profile Pic* |
| *DB5* | *Medium* | *F* | *Tim Durkee*  *(Dev Team)* | *U10* | *Edit Profile (Coach)* |
| *DB6* | *High* | *F* | *Tim Durkee*  *(Dev Team)* | *U11* | *Updated date Password* |
| *DB7* | *Low* | *F* | *Tim Durkee*  *(Dev Team)* | *U13* | *View Roster via Sessions* |
| *DB8* | *High* | *F* | *Adam Smith*  *(Dev Team)* | *U11, U16* | *All passwords must be encrypted using a salted hash technique.* |

***ID*** *– Unique requirement ID*

***Priority*** *– Priority of this requirement*

***Type*** *– Functional(F) or Non-functional(NF)*

***Source*** *– Who is most interested in this requirement (John Smith – Customer). For this project you can make it up, in reality you’ll want to capture this as you capture the requirements.*

***Contained in Use Case(s****) – Which use cases reference this requirement or which use cases when executed will perform this requirement. There may be a few functional requirements without a use-case and the non-functional requirements generally will NOT be part of a use-case (so put N/A).*

***Description*** *– The description of the requirement. “The system shall …. “*

*These requirements should match up with your use case diagrams.*

**Appendix A: Analysis Models**

**Don’t do any of these for CS421 SRS. You will create these models during the high level design deliverable.**

**Appendix B: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*

*List here any open questions or things you know still need to be done to the SRS, but haven’t been addressed yet. (It’s okay to have things like that, especially in this project because we don’t have time to do everything.)*