**SSD: Register to System (as Client)**

* Contract: LoginForm (as Client)
  + Operation: LoginForm(userType: “client”)
  + Cross References: Use Case: Register to System, Login to System
  + Preconditions:
    - User clicked on “Client login”
  + Postconditions:
    - A LoginForm(“client”) is created (instance creation)
      * i.e. User directed to the Client login form
* Contract: CreateAccountClient
  + Operation: CreateAccountClient()
  + Cross References: Use Case: Register to System
  + Preconditions:
    - User clicked on “Create Account as Client” after clicking on “Client Login”
  + Postconditions:
    - A CreateAccountClient is created (instance creation)
      * i.e. User directed to the Client register form
* Contract: registerClient
  + Operation: registerClient(name: String, phoneNumber: int, age: int)
  + Cross References: Use Case: Register to System
  + Preconditions:
    - User entered valid name, phoneNumber and age
  + Postconditions:
    - A new Client c is created (instance creation)
      * c is added to the database
    - A new ClientPage(c) is created (instance creation)
      * i.e. User directed to Client Page, with the new Client’s information

**SSD: Register to System (as Instructor)**

* Contract: LoginForm (as Instructor)
  + Operation: LoginForm(userType: “instructor”)
  + Cross References: Use Case: Register to System, Login to System
  + Preconditions:
    - User clicked on “Instructor login”
  + Postconditions:
    - A LoginForm(“instructor”) is created (instance creation)
      * i.e. User directed to the Instructor login form
* Contract: CreateAccountInstructor
  + Operation: CreateAccountInstructor()
  + Cross References: Use Case: Register to System
  + Preconditions:
    - User clicked on “Create Account as Instructor” after clicking on “Instructor Login”
  + Postconditions:
    - A CreateAccountInstructor is created (instance creation)
      * i.e. User directed to the Instructor register form
* Contract: registerInstructor
  + Operation: registerInstructor(name: String, phoneNumber: int, specialty: String, cities: String)
  + Cross References: Use Case: Register to System
  + Preconditions:
    - User entered valid name, phoneNumber, specialty and cities
  + Postconditions:
    - A new Instructor i is created (instance creation)
      * i is added to the database
    - A new InstructorPage(i) is created (instance creation)
      * i.e. User directed to Instructor Page, with the new Instructor’s information

**SSD: Login to System (as Client)**

* Contract: loginClient
  + Operation: loginClient(name: String, phoneNumber: int)
  + Cross References: Use Case: Login to System
  + Preconditions:
    - User entered client credentials (name and phoneNumber) that exist in the Database
  + Postconditions:
    - A new Client c is created (instance creation)
    - A new ClientPage(c) is created (instance creation)
      * i.e. User directed to Client Page, with their Client information

**SSD: Login to System (as Instructor)**

* Contract: loginInstructor
  + Operation: loginInstructor(name: String, phoneNumber: int)
  + Cross References: Use Case: Login to System
  + Preconditions:
    - User entered instructor credentials (name and phoneNumber) that exist in the Database
  + Postconditions:
    - A new Instructor i is created (instance creation)
    - A new InstructorPage(i) is created (instance creation)
      * i.e. User directed to Instructor Page, with their Instructor information

**Use Case: Login to System (as Admin)**

* Contract: LoginForm (as Admin)
  + Operation: LoginForm(userType: “admin”)
  + Cross References: Use Case: Register to System, Login to System
  + Preconditions:
    - User clicked on “Admin login”
  + Postconditions:
    - A LoginForm(“admin”) is created (instance creation)
      * i.e. User directed to the Admin login form
* Contract: loginAdmin
  + Operation: loginAdmin(name: String, password: String)
  + Cross References: Use Case: Login to System
  + Preconditions:
    - User entered the correct Admin credentials (name and password)
  + Postconditions:
    - The Admin instance is returned (Admin is a Singleton)
    - A new AdminPage(a) is created (instance creation)
      * i.e. User directed to Admin Page

**Use Case: View all Available Offerings (as Admin)**

* Contract: ViewOfferingsPage (as Admin)
  + Operation: ViewOfferingsPage(Actor: admin)
  + Cross References: Use Case: View all Available Offerings, View Instructor-Assigned Offerings
  + Preconditions:
    - User has gone through the “Login to System (as Admin)” use case successfully
    - User clicked on “View all offerings”
  + Postconditions:
    - * A new ViewOfferingsPage(admin) is created (instance creation)
        + For Admins, ViewOfferingsPage will have a table with all offerings that are NOT at full capacity, regardless of whether they have been assigned to an instructor or not
        + An Admin will have various options that are appropriate for Admin operations, such as delete offering or edit offering

**Use Case: View all Available Offerings (as Instructor)**

* Contract: ViewOfferingsPage (as Instructor)
  + Operation: ViewOfferingsPage(Actor: instructor)
  + Cross References: Use Case: View all Available Offerings, View Instructor-Assigned Offerings
  + Preconditions:
    - User has gone through either the “Register to System (as Instructor)” or “Login to System (as Instructor)” use cases successfully
    - User clicked on “View all offerings”
  + Postconditions:
    - * A new ViewOfferingsPage(instructor) is created (instance creation)
        + For Instructors, ViewOfferingsPage will have a table with all offerings that are NOT at full capacity, regardless of whether they have been assigned to an instructor or not
        + An instructor will have the option to take on Offerings that have not already been assigned to an instructor

**Use Case: View Instructor-Assigned Offerings**

* Contract: ViewOfferingsPage (as Client)
  + Operation: ViewOfferingsPage(Actor: client)
  + Cross References: Use Case: View all Available Offerings, View Instructor-Assigned Offerings
  + Preconditions:
    - User has gone through either the “Register to System (as Client)” or “Login to System (as Client)” use cases successfully
    - User clicked on “View all offerings”
  + Postconditions:
    - * A new ViewOfferingsPage(client) is created (instance creation)
        + For a Client specifically, ViewOfferingsPage will have a table with all offerings that have been assigned to an instructor
        + Offerings that are at full capacity (i.e. bookings) will be visible with limited details
        + A Client will have the option to reserve Offerings that have been assigned to an instructor and are not at full capacity

**SSD: Instructor Taking Offering**

* Contract: takeOffering
  + Operation: takeOffering(Offering: offering)
  + Cross References: Use Case: Take on offerings
  + Preconditions:
    - User has gone through either the “Register to System (as Instructor)” or “Login to System (as Instructor)” use cases successfully
    - User clicked on “View all offerings”
    - User has clicked on the option to take on a particular Offering
    - The Offering does not already have an Instructor associated with it, and it is not at full capacity
  + Postconditions:
    - The Offering is associated to the instructor (association formed)
      * In the DB, the Offering has a column for the Instructor’s ID, which will now contain this Instructor’s ID
    - A confirmation message is shown to the user

**SSD: Admin Enter Organization Offering**

* Contract: EnterNewOfferingPage
  + Operation: EnterNewOfferingPage (Admin: admin)
  + Cross References: Use Case: Enter Organization’s Offerings
  + Preconditions:
    - User has gone through the “Login to System (as Admin)” use case successfully
    - User clicked on “EnterNewOfferingPage”
  + Postconditions:
    - A new EnterNewOfferingPage(admin) is created (instance creation)
      * i.e. User directed to a form to enter all required Offering information
* Contract: enterNewOffering
  + Operation: enterNewOffering(city, location, capacity, startTime, endTime, classType)
  + Cross References: Use Case: Enter Organization’s Offerings
  + Preconditions:
    - User has gone through EnterNewOfferingPage(Admin: admin) operation
    - The inputs for city, location, capacity, startTime, endTime and classType are all valid according to the logic of the requirements
  + Postconditions:
    - An Offering with the specified parameters (city, etc.) is added to the DB (Instance creation)
      * Note: The Offering has no associated Instructor nor Clients yet
    - A confirmation message is shown to the user