Design Document

for

SISCalendar

Prepared by Zach Masiello
Ethan Mick
Michael Caputo
Shawn Thompson
Organization: SIS.io

Table of Contents

1. Introduction
2. System Overview
    2.1 High Level Description
    2.2 Technology Stack
3. Technical Approach
    3.1. Tools
4. System Architecture
    3.1 High Level Architecture
    3.2 Deployment
    3.3 SubSystem Design
5. Class Diagrams
    5.1. Schedule Subsystem Diagrams
    5.2. SISCalendar application flow
    5.3. Authentication Subsystem Diagrams
6. Sequence Diagrams
    6.1 iCal
    6.2 Log in

Revision History

Name	Date	Reason For Changes	Version
Initial	10/16/13	First Version	1.0
Revision 1	10/29/13	Updated Class Diagrams	1.1
Revision 2	04/2/14	Updated diagrams	1.2
Revision 3	04/12/14	Updated diagrams based on code review feedback	1.3

1. Introduction

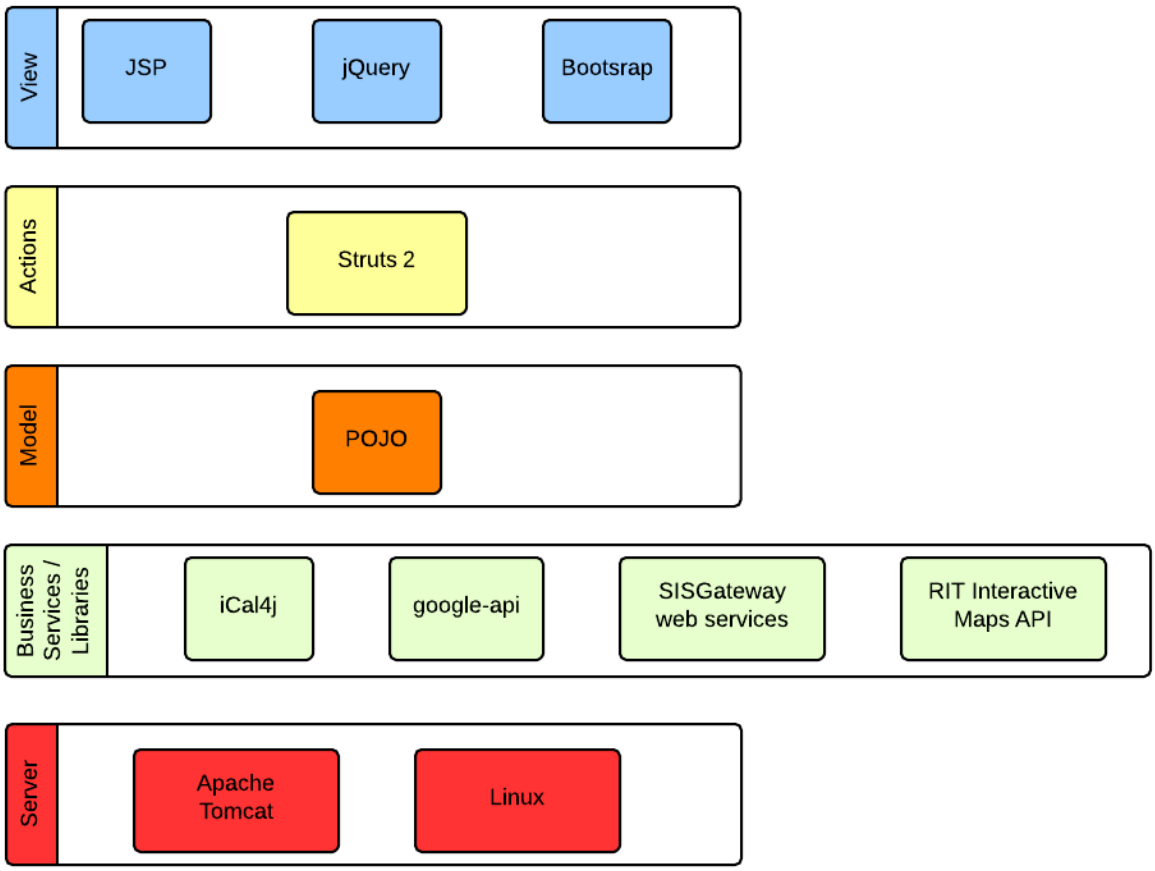
The purpose of this document is to outline the design for SISCalendar. This will include a view of the highlevel architecture as well as the breakdown of the internal subsystems. UML class and sequence diagrams will be provided to show how the system will be put together and how data will flow through the system. There will also be discussion on the technologies that we will be using throughout this project. This document provides an outline of the user interface to demonstrate how it will be formatted. Additionally, there is a section that makes use of a requirements traceability matrix, which will make it easier to trace system features and designs back to the requirements.

2. System Overview

2.1 High Level Description

text

2.2 Technology Stack



View

All the tools and technologies used to render the view for the client side of SISCalendar

JSP - Java Server Pages. High level abstraction of java servlets that can be cached, changed, and rendered at runtime.
JQuery - A javascript library that allows for easier manipulation of the Document Object Model, simpler AJAX calls, and animations.
Bootstrap - A CSS library that allows easy templating of websites and includes built in responsiveness.

Actions

All the tools and technologies that handle the communication between the model and view layers.

Struts 2 - An MVC framework that allows simple creation of enterprise-grade web applications. The “controller” item in Struts 2 is an action, an out of the box class that can be used to send information to the view to be rendered.

Model

All the tools and technologies that are used to store data needed by SISCalendar to operate.

POJO - Plain Old Java Object. SISCalendar stores all of its data (Course information, exam information, user data) as java objects that can be easily passed to and manipulated by the Actions layer.

Business Services / Libraries

All of the external applications and libraries that SISCalendar utilizes.

iCal4j - A java library that transforms java objects into iCal format, and can create an iCal file.
google-api - Google's custom API that handles integration with any of their services. SISCalendar relies on this library to automatically upload class information to a student's Google Calendar.
SISGateway Web Services - The API that SISCalendar uses to access all of the course information related to a students account.
RIT Interactive Map - RIT's API that allows information to be displayed in Google Maps with an overlay over RIT. SISCalendar uses this information to display the location of classes as well as display information about various buildings.

Server

All the tools and technologies that are installed on the servers running SISCalendar

Apache Tomcat - The web server that is used to run the SISCalendar application.
Linux - The OS of the server that SISCalendar is hosted on.
ANT - A Java library used to make the build and deployment process easier. Creates war files from the struts application.

3. Technical Approach

3.1. Tools

There are a number of tools and technologies that this project will utilize for development.

- Server Side
    - Java 1.7.0_40 environment
    - Apache Struts 2.3.15.2 framework
    - Apache ANT
    - Eclipse Kepler 4.3 IDE
    - Git repository (on Github)

- Client Side
    - Jquery 2.0.3
    - normalize.css

We have decided to use the Struts 2 framework for our development for a number of reasons. Struts 2 is the most up-to-date (stable) version of the Struts framework currently released by Apache (updated as of 09-21-13). This means that problems stemming from the framework itself should be minimal. Struts is also one of the approved technologies given to the student team as it is currently already supported by ITS in the Team Builder project. The Team Builder project has already solved the problem of Shibbeloth authentication in the Struts 2 framework, which will allow the SIS.io team to reuse code shortening the expected schedule and avoiding unforseen challenges caused by Shibbeloth authentication with new technologies.

Struts 2 also provides a Model-View-Controller paradigm to applications developed in it. This will allow the SIS.io team to create a clean, reusable, and easily extendable application within the confines of the schedule.

Apache Ant will be used in the creation of .war file builds for development, testing, and production. Ant is one of the ITS approved libraries for creating java builds for deployment, as well as being the library used by the Team Builder application. SIS.io will be able to utilize Team Builders Ant scripts as examples when constructing our own, making the process as a whole easier.

Development for SISCalendar will take place on each students own personal computer. Eclipse Kepler 4.3 shall be used as the integrated development environment. Eclipse was chosen because it has plugins that will allow easy integration for both the Struts framework and also Apache Ant to aid in the development and deployment processes.

SIS.io will use a git repository to store our code. A master branch will be stored on the account provided by RIT ITS, and each SIS.io developer will maintain their own local copy and feature branches.

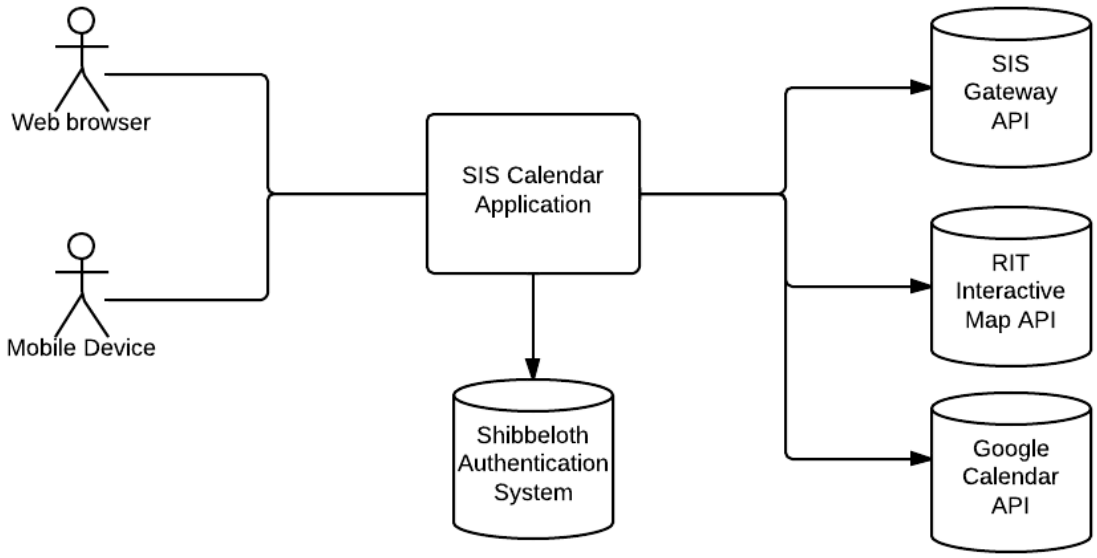
For rendering and interacting with the client side, SISCalendar will utilize JQuery and normalize.css. JQuery will allow the manipulation of data and UI elements easily and efficiently on the client side. Normalize.css will be used to ensure SISCalendar has cross-browser consistency when the UI is rendered.

4. System Architecture

3.1 High Level Architecture

The SISCalendar application will interact with several major components. These components will be the Shibbeloth Authentication System (used to authenticate and authorize a user before

attempting to download a calendar), the SIS Gateway API (used to retrieve all calendar data for a given user), and the RIT Interactive Map API (used to generate a clickable URL to display a specific building on RIT's campus). The application will also interact with Google's Calendar API to allow students to automatically load their class schedule into their Google calendar.



3.2 Deployment

Deployment of the SISCalendar project will follow the precedent set by the Team Builder project. War files will be built using ANT scripts to make deployment to different environments as simple as possible.

To deploy wiithout SVN
1. Run the ant build script - Hopefully you don't run into problems with this step
2. scp the file. It will be dist/sisCalendar.war to <username>@siscalendardev.rit.edu:~/
3. ssh into <username>@siscalendardev.rit.edu
Run these commands:
4. sudo su - webapps
5. cp /home/<username>/sisCalendar.war ~/staging/sisCalendar.war
6. bin/cluster/chgMgmt/deploySisCalendar

To deploy wiith SVN
1. Run the ant build script
2. copy the dist/sisCalendar.war file into our svn in the trunk/dev/release/ folder
3. use the command svn ci -m "<CHECKIN MESSAGE>"
3. ssh into <username>@siscalendardev.rit.edu
Run These commands:

4. sudo su - webapps
5. bin/cluster/chgMgmt/getReleasedSisCalendar
6. bin/cluster/chgMgmt/deploySisCalendar

the logs are at: ~/instances/sisCalendarCluster/node1818/logs/catalina.out

This is the command used to view the logs:
tail -f -n 100 ~/instances/sisCalendarCluster/node1818/logs/catalina.out

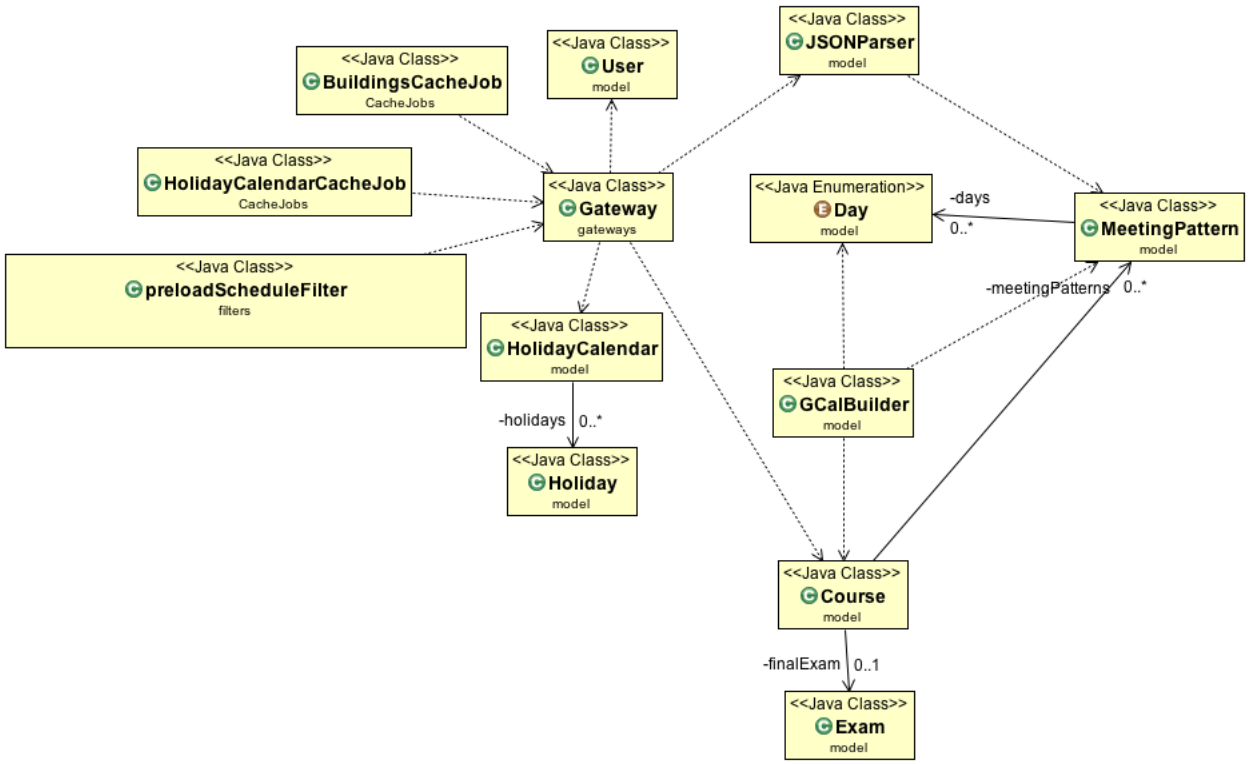
3.3 SubSystem Design

There will be 2 main subsystems in this application. Those subsystems are the Authentication subsystem, and the Schedule subsystem.

5. Class Diagrams

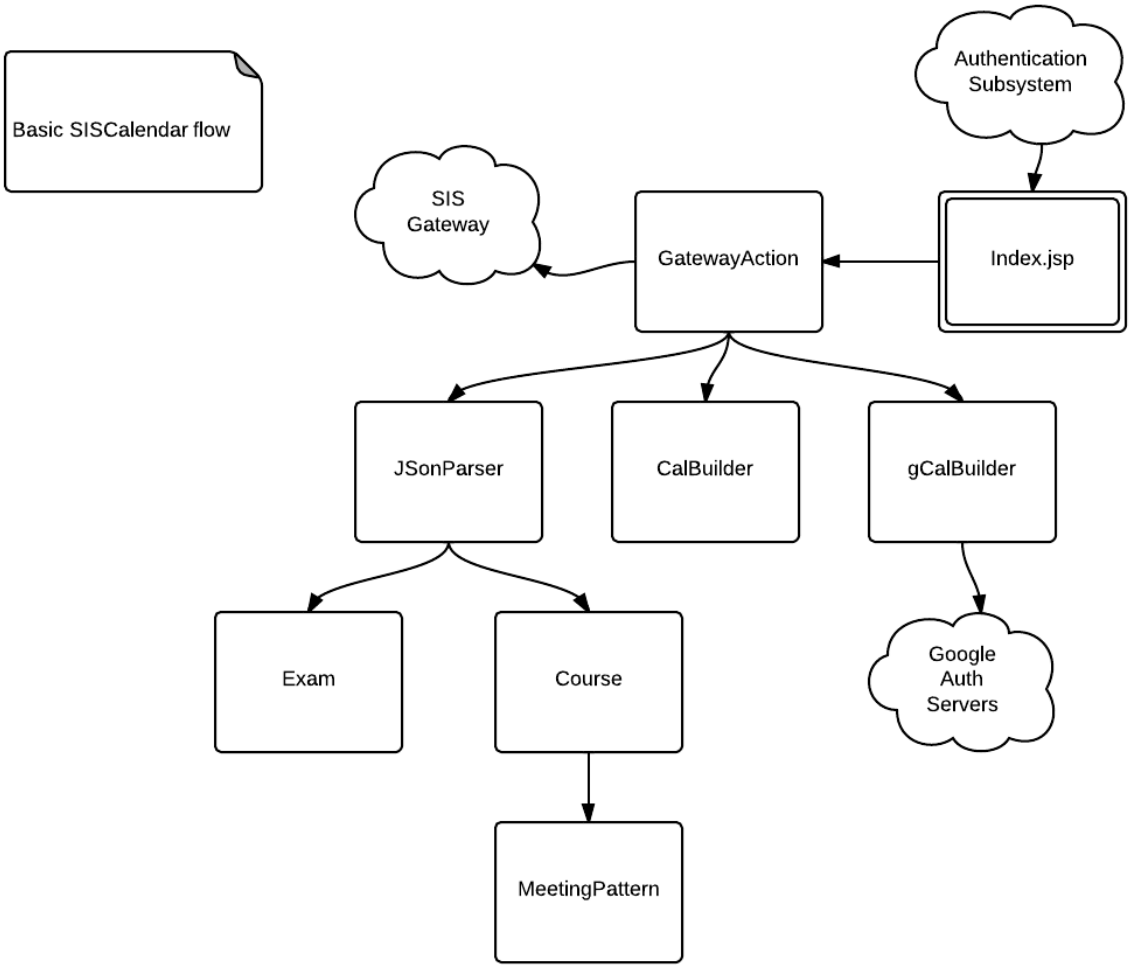
5.1. Schedule Subsystem Diagrams

Class diagram for the main Schedule subsystem. This is the system responsible for retrieving class and map information and manipulating that data to something usable by the user.



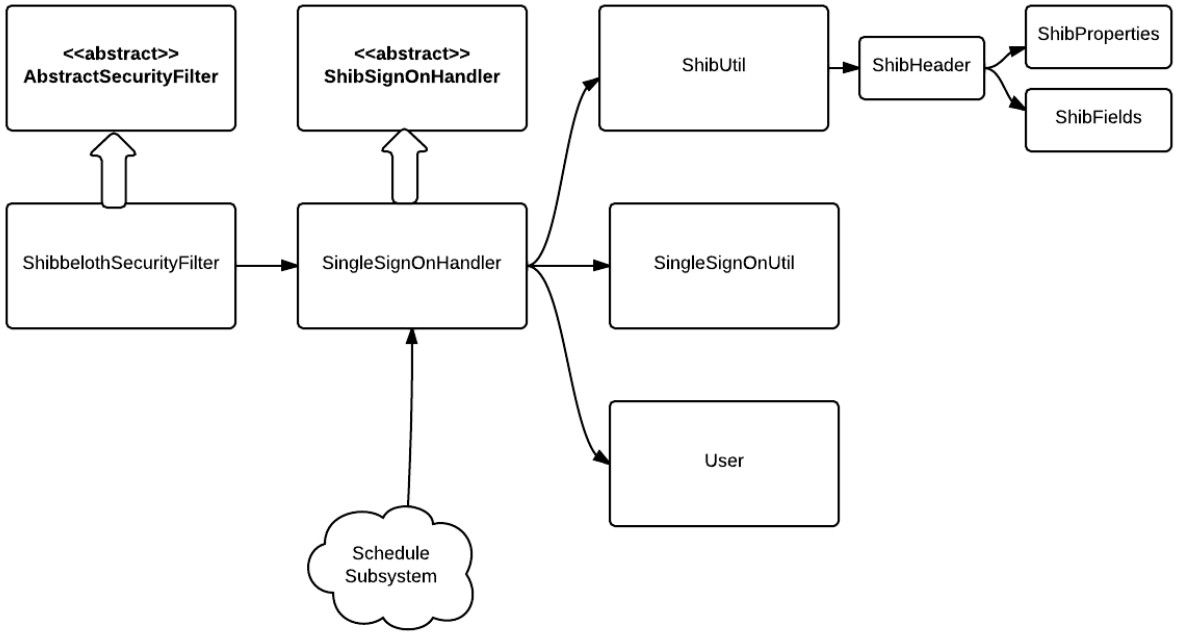
5.2. SISCalendar application flow

This diagram shows the basic flow of events within the SISCalendar application. Note that a user will initially enter from the Authentication subsystem, reaching Index.jsp once they are properly authenticated.



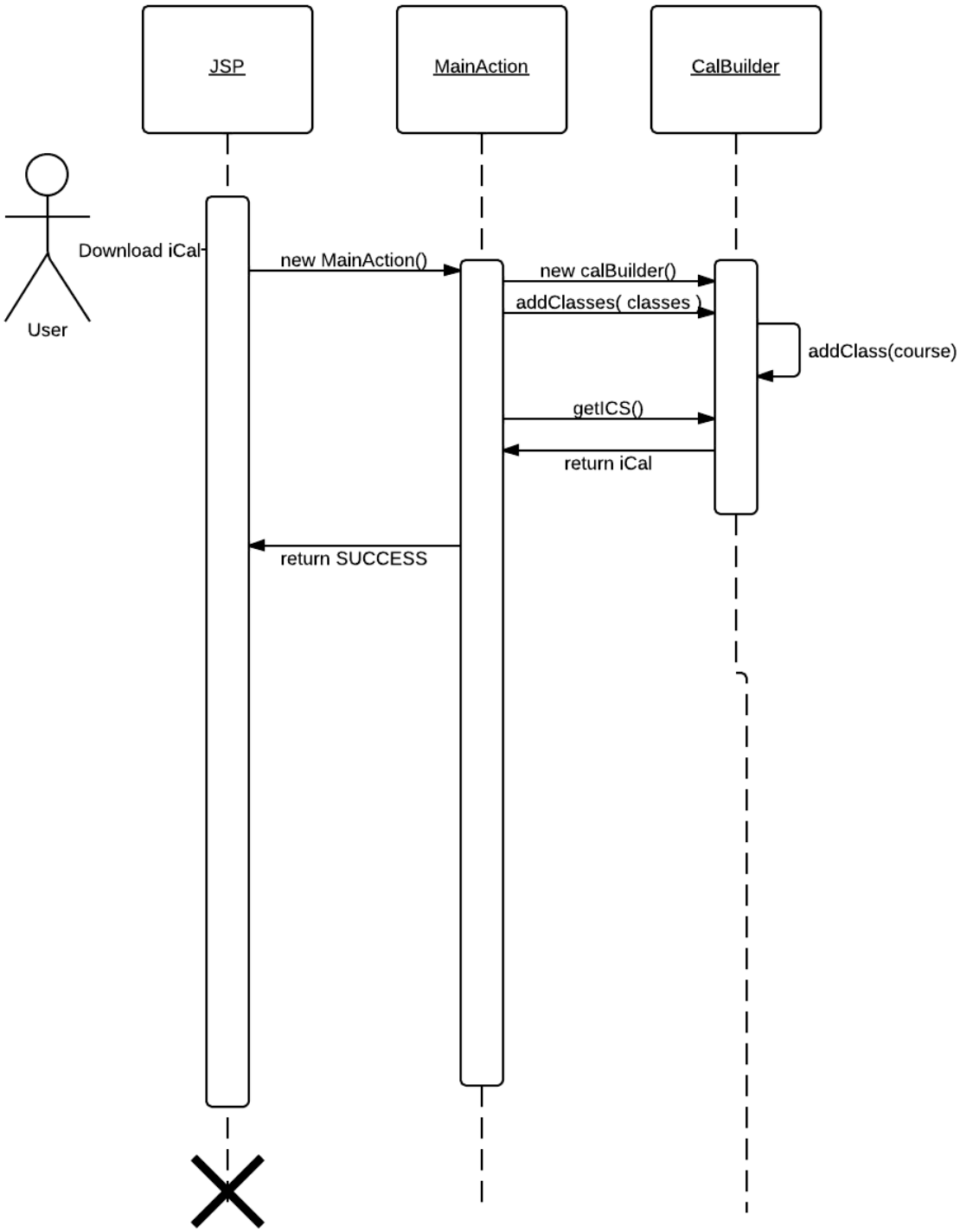
5.3. Authentication Subsystem Diagrams

Initial class diagram of the authentication subsystem. This is based off of the authentication system put into place and currently used by the TeamBuilder project.

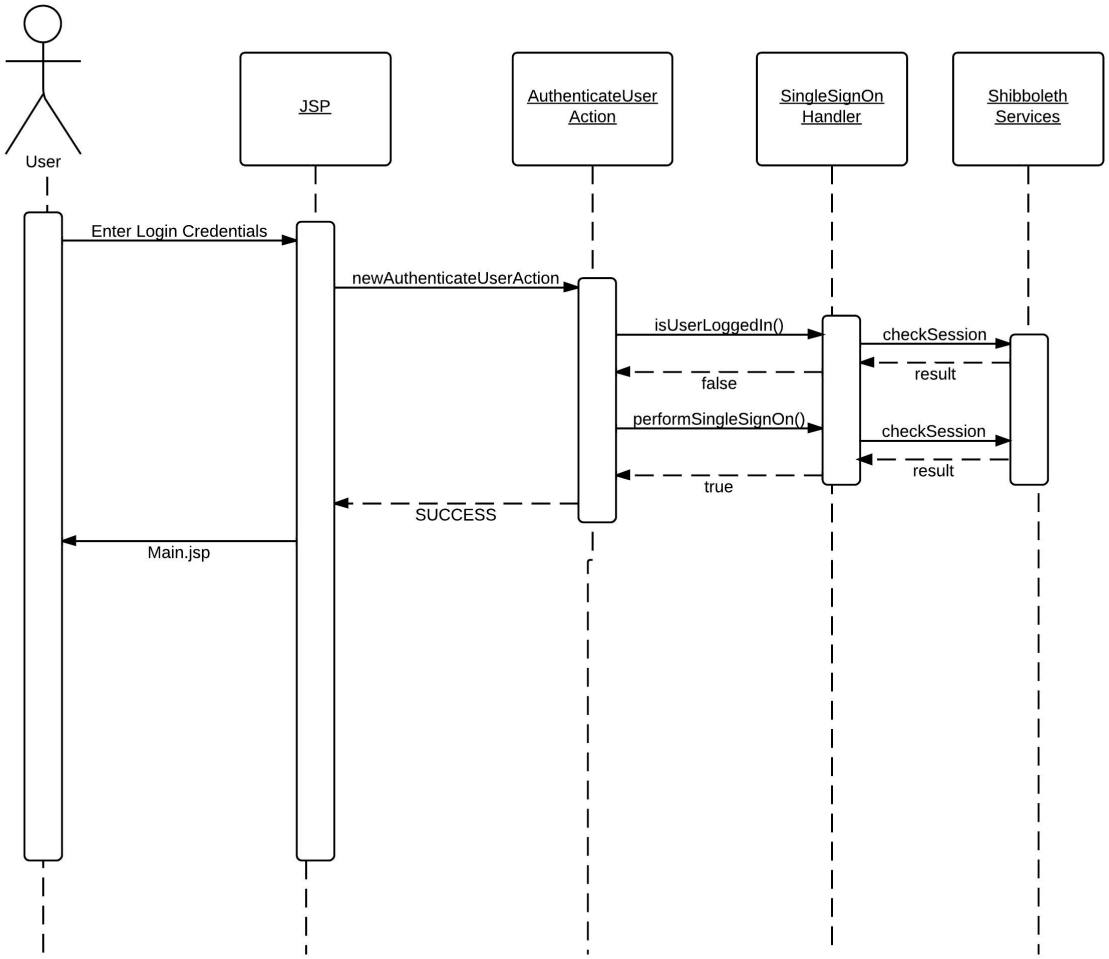


6. Sequence Diagrams

6.1 iCal


JSP
MainAction
CalBuilder
User
Download iCal
new MainAction()
new calBuilder()
addClasses(classes)
addClass(course)
getICS()
return iCal
return SUCCESS

6.2 Log in


User
JSP
AuthenticateUser Action
SingleSignOn Handler
Shibboleth Services
Enter Login Credentials
newAuthenticateUserAction
isUserLoggedIn()
checkSession
result
false
performSingleSignOn()
checkSession
result
true
SUCCESS
Main.jsp

6.3 Fetch Schedule