Hi Claudia! I turned on track changes so you can see the things I changed. Just click the vertical red bars on the left. Let me know if you have any questions ^\_^ --Erin

# Introduction

The following document covers the testing involved in the system. A full breakdown of each test case is provided, accompanied by the results. This is useful information in the installation procedure of this software for any users, which is shown alongside the user manual of this section. The types of testing includes unit testing, requirement testing, stress testing and security testing. The final topic included is the final cost estimation of the entire project.

# Project Description

The proposed and outlined web application, known as TimeTurner, is designed to auto-generate a student’s course sequence from their first semester up until the end of their degree. It takes into account user input preferences and and any previously completed courses or course prerequisites before creating this sequence. Preferences can be made by the student and include options such as night classes or having particular days off. The application will notify the user if a certain preference suggested results in an impossibility or conflict in the sequence. This sequence generator will be able to create a sequence at any point throughout the user’s degree, if sudden change in circumstances were to arise.

The goal of this application is to simplify the method with which students may decide and schedule their courses. If a course must be redone, the generator can decide what other courses should move where in regards to the remaining courses to be completed, which can be done in seconds, rather than hours. It saves the time of the user, in a simple and efficient manner. Ultimately, the system’s end goal will be to simplify a student’s task of creating their own course schedule in order to allow students to redirect their time to other more important activities, thus making course registration much simpler, quicker, and easier.

# Testing Report

## Test Coverage

While it is ideal to test every single item and every possible requirement of a system, this is often impossible due to time and budget constraints. In this section, we describe which items of the system were tested, why it was important to test them and which tools were used for testing. We also describe which items were not tested, yet would have been tested given more time and/or money.

Tested Items

Unit Testing:

Unit testing focuses on testing small units of code at a time. It breaks the system down into its individual classes or methods and tests them each separately. Unit testing is important for making sure each piece of an application works as expected, so that these pieces can then be combined with other parts of the application in order to ultimately have a full working system.

Here, Unit testing was done using PHPUnit 5.3. We performed unit tests on two of the model classes in our system, described below:

Tested items:

* **LoginForm:** it was important to test the LoginForm class because this class is needed for users to access the system. Without access to the system, nothing else can be done. To test this unit, we created a PHPUnit test class called LoginFormTest, which consists of methods to test each function of the LoginForm class, as well as a setUp() method that creates a testable LoginForm object and provides sample input (username and password) for the form.
* **UserSchedule:** the UserSchedule model class was chosen for unit testing since it accesses the database and is critical for generating a schedule. To test this unit, we again made a PHPUnit test class, called UserScheduleTest with methods to test each function. The setUp() method creates a testable UserSchedule object and opens a connection to the database, and the tearDown() method closes the database connection after the test is run.

Requirements Testing:

Requirements testing breaks a system down based on its requirements, and tests each one of them to ensure they function as expected. It is important for checking which requirements are met and which are not met. For each different requirement, several test cases are performed with different possible scenarios or inputs. Here, each test case includes a test ID, description (describing the input), expected output, actual output and result. If the actual output matches the expected output, the result is a *pass*. If not, the result is a *fail*. For any failed tests, we can return to the source code used to implement this requirement and identify the bugs that caused it to fail. We tested the following requirements:

Tested items:

* **UC1:** Login
* **UC2:** Logout
* **UC4:** Browse Course List
* **UC6:** Generate Schedule
* **UC7:** View Saved Schedules
* **UC8:** View Academic Record
* **UC9:** Drop Course
* **UC11:** Save Generated Schedule
* **UC13:** Manage Courses
* **UC14:** Create Course
* **UC15:** View Courses
* **UC16:** Add User
* **UC17:** Search Users
* **UC18:** View User
* **UC19:** Update User
* **UC20:** Delete User
* **UC21:** Browse Users

Stress Testing:

Stress testing is used to test an application’s performance under extreme conditions of usage, such as a high volume of users or a large data load. It is done in order to determine the system’s robustness and reliability, and helps to identify issues that may only arise under stressful conditions.

There are a number of tools available for stress testing a web application that simulate extreme conditions. Here, we used Apache JMeter, a Java application that works by creating multiple threads that simultaneously access a given web server via HTTP requests. We tested stress conditions for both the login and schedule generation functions of the system.

Security Testing:

Security testing is crucial for any software system to ensure that the system can protect itself against malicious attacks. We used two tools to test security. The first was SQL Inject Me, a Firefox plugin that sends SQL injection strings through the forms of a webpage and checks for any database errors that may occur as a result. The second was Nikto, a web server scanner that uses a Perl script to perform multiple tests against a given server, including checking for dangerous files and checking for outdated versions.

Untested Items of Interest

For the unit tests, we were only able to test 2 of the 11 models in our system. This was due to tight time constraints, as it took a while to set up PHPUnit and write the test code. We chose to test the LoginForm and UserSchedule models because these classes are very important to the system’s functionality; every user needs the LoginForm to access the system, and UserSchedule is needed to generate a schedule. With more time, we would have ideally performed unit tests on the other 9 models in our system as well as the 5 controller classes, listed below:

Untested units:

* Model classes:
  + CompletedCourses
  + ContactForm
  + Course
  + PreferenceForm
  + Prerequisite
  + Section
  + Subsection
  + User
  + UserSchedules
* Controller classes:
  + CourseController
  + ProfileController
  + SchedulerController
  + SiteController
  + UserController

For the requirements tests, we tested all requirements except the ones that were scoped out. The following requirements were not tested:

Untested requirements:

* **UC3:** Create Course Sequence
* **UC5:** View Course Sequence
* **UC10:** Add Course
* **UC12:** View Weekly Schedule

## Test Cases

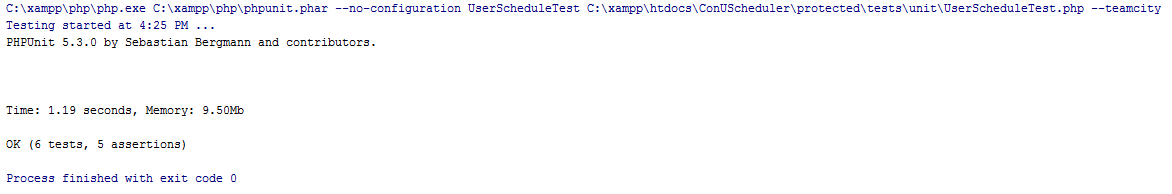
Unit Testing

Code for UserScheduleTest:

**<?php  
require\_once**(**'../../../protected/models/UserSchedule.php'**);  
$yiit=**'../../../framework/yiit.php'**;  
**require\_once**($yiit);  
  
**class** UserScheduleTest **extends** CDbTestCase {  
 **protected $fixture**;  
 **protected $db**;  
  
 **protected function** setUp() {  
 $this->**db** = **new** CDbConnection(**'mysql:host=conuscheduler.ddns.net;dbname=soen341'**,**'soen341'**,**'soen341'**);  
 $this->**db**->**active**=**true**;  
 CActiveRecord::*$db*=$this->**db**;  
 $this->**fixture** = **new** UserSchedule;  
 }  
  
 **protected function** tearDown() {  
 **if**($this->**db**)  
 $this->**db**->**active**=**false**;  
 }  
  
 **public function** testTableName() {  
 $expected = **'user\_schedule'**;  
 $results = $this->**fixture**->tableName();  
 $this->assertEquals($expected,$results);  
 }  
  
 **public function** testRules() {  
 $expected = **array**(  
 **array**(**'scheduleID, courseID, sectionID, subsectionID, year'**, **'required'**),  
 **array**(**'scheduleID, courseID, sectionID, subsectionID, year'**, **'numerical'**, **'integerOnly'**=>**true**),  
 **array**(**'ID, scheduleID, courseID, sectionID, subsectionID, year'**, **'safe'**, **'on'**=>**'search'**),  
 );  
 $results = $this->**fixture**->rules();  
 $this->assertEquals($expected,$results);  
 }  
  
 **public function** testRelations() {  
  
 $expected = **array**(  
 **'scheduleID'** => **array**(UserSchedule::***BELONGS\_TO***, **'UserSchedules'**, **'ID'**)  
 );  
 $results = $this->**fixture**->relations();  
 $this->assertEquals($expected,$results);  
 }  
  
 **public function** testAttributeLabels() {  
 $expected = **array**(  
 **'ID'** => **'ID'**,  
 **'scheduleID'** => **'Schedule'**,  
 **'courseID'** => **'Course'**,  
 **'sectionID'** => **'Section'**,  
 **'subsectionID'** => **'Subsection'**,  
 **'year'** => **'Year'**,  
 );  
 $results = $this->**fixture**->attributeLabels();  
 $this->assertEquals($expected, $results);  
 }  
  
 **public function** testSearch() {  
 $stub = $this->getMockBuilder(**'CDbCriteria'**)  
 ->disableOriginalConstructor()  
 ->getMock();  
 $stub->expects($this->any())  
 ->method(**'compare'**)  
 ->will($this->returnValue(**'true'**));  
 }  
  
 **public function** testModel() {  
 $stub = $this->getMockBuilder(**'UserSchedule'**)  
 ->disableOriginalConstructor()  
 ->getMock();  
 $stub->expects($this->any())  
 ->method(**'model'**)  
 ->will($this->returnValue($this->**fixture**));  
 $this->assertNotNull($stub);  
 }  
}

**Results:**

UserScheduleTest passed 6/6 tests, shown in the figure below:

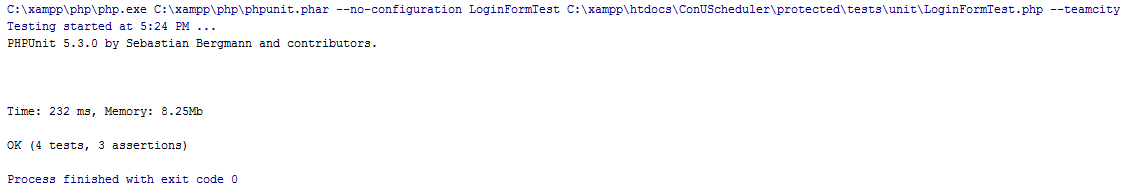


Code for LoginForm test:

**<?php  
  
require\_once**(**'../../../protected/models/LoginForm.php'**);  
**require\_once**(**'../../../protected/components/UserIdentity.php'**);  
$yiit=**'../../../framework/yiit.php'**;  
**require\_once**($yiit);  
  
**class** LoginFormTest **extends** CTestCase  
{  
 **protected $fixture**;  
 **protected function** setUp() {  
 $this->**fixture** = **new** LoginForm;  
 $this->**fixture**->**username** = **"erin"**;  
 $this->**fixture**->**password** = **"soen341"**;  
 $this->**fixture**->**rememberMe** = **"0"**;  
 }  
  
 **protected function** tearDown() {  
 }  
  
 **public function** testRules() {  
 $stub = **array**(**array**(**'username, password'**, **'required'**),  
 **array**(**'rememberMe'**, **'boolean'**),  
 **array**(**'password'**, **'authenticate'**));  
 $results = $this->**fixture**->rules();  
 $this->assertEquals($stub,$results);  
 }  
  
 **public function** testAttributeLabels() {  
 $stub = **array**(**'rememberMe'**=>**'Remember me next time'**,);  
 $results = $this->**fixture**->attributeLabels();  
  
 $this->assertEquals($stub, $results);  
 }  
  
 **public function** testAuthenticate() {  
 $stub = $this->getMockBuilder(**'UserIdentity'**)  
 ->disableOriginalConstructor()  
 ->getMock();  
 $stub->expects($this->any())  
 ->method(**'authenticate'**)  
 ->will($this->returnValue(**'true'**));  
 }  
  
 **public function** testLogin\_2() {  
 $stub = $this->getMockBuilder(**'LoginForm'**)  
 ->disableOriginalConstructor()  
 ->getMock();  
 $stub->expects($this->any())  
 ->method(**'login'**)  
 ->will($this->returnValue(**'true'**));  
 $this->assertEquals(**'true'**, $stub->login());  
 }  
}

**Results:**

LoginFormTest passed 4/4 tests, shown in the figure below:



Requirements Testing

Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC1** | | **Login (Administrator)** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 1.1 | Username: Admin  Password: 1234 | | Home Page | Home Page | **Pass** |
| 1.2 | Username: Admine  Password: 1234 | | Incorrect Username  or Password Error | Incorrect Username  or Password Error | **Pass** |
| 1.3 | Username: Admin  Password: 12345 | | Incorrect Username  or Password Error | Incorrect Username  or Password Error | **Pass** |
| 1.4 | Username: Admine  Password: 12345 | | Incorrect Username  or Password Error | Incorrect Username  or Password Error | **Pass** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC1** | | **Login (Student)** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 1.5 | Username: Erin  Password: soen341 | | Home Page | Home Page | **Pass** |
| 1.6 | Username: Erine  Password: soen341 | | Incorrect Username  or Password Error | Incorrect Username  or Password Error | **Pass** |
| 1.7 | Username: Erin  Password: soen341 | | Incorrect Username  or Password Error | Incorrect Username  or Password Error | **Pass** |
| 1.8 | Username: Erine  Password: soen341 | | Incorrect Username  or Password Error | Incorrect Username  or Password Error | **Pass** |

Logout

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC2** | | **Logout (Administrator)** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 2.1 | User clicks logout tab | | Home Page | Home Page | **Pass** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC2** | | **Logout (Administrator)** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 2.2 | User clicks logout tab | | Home Page | Home Page | **Pass** |

Create Course Sequence

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~UC3~~** | | **~~Create Course Sequence~~** | | | |
| ~~Test ID~~ | ~~Description~~ | | ~~Expected Output~~ | ~~Actual Output~~ | ~~Result~~ |

Browse Course List

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC4** | | **Browse Course List** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 4.1 | Student requests to view course list | | List of all courses in the system is displayed | List of all courses in the system is displayed | **Pass** |
| 4.2 | Student clicks on a prerequisite course | | Course information for that course is displayed | Course information for that course is displayed | **Pass** |

View Course Sequence

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~UC5~~** | | **~~View Course Sequence~~** | | | |
| ~~Test ID~~ | ~~Description~~ | | ~~Expected Output~~ | ~~Actual Output~~ | ~~Result~~ |
| ~~5.1~~ | ~~Student requests to view course sequence with no saved schedule~~ | | ~~Course sequence is displayed~~ |  |  |
| ~~5.2~~ | ~~Student requests to view course sequence after adding a course~~ | | ~~Course sequence is displayed~~ |  |  |
| ~~5.3~~ | ~~Student requests to view course sequence after dropping a course~~ | | ~~Course sequence is displayed~~ |  |  |
| ~~5.4~~ | ~~Student requests to view course sequence after dropping all courses~~ | | ~~Course sequence is displayed~~ |  |  |

Generate Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC6** | | **Generate Schedule** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 6.1 | Student leaves the preferences blank and generates the schedule for a given year. | | All courses for each semester in each academic year selected is displayed. | All courses for each semester in each academic year selected is displayed. | **Pass** |
| 6.2 | Student selects a time preferences without selecting the day checkbox, then generates schedule for a given year. | | All courses for each semester in each academic year selected is displayed. | All courses for each semester in each academic year selected is displayed. | **Pass** |
| 6.3 | Student selects a day without specifying the time preferences for that day, then generates a schedule for a given year. | | No courses are displayed: ‘no result’ | No courses are displayed: ‘no result’ | **Pass** |
| 6.4 | Student selects a day and the times for that day as a preference, and generates the schedule for a given year. | | Only courses that match the time filter are shown, no result found if no classes match the time criteria. | Only courses that match the time filter are shown, no result found if no classes match the time criteria. | **Pass** |
| 6.5 | Student selects a set of time preferences, but no year and generates the schedule. | | No courses are displayed: ‘no result’. | No courses are displayed: ‘no result’. | **Pass** |
| 6.6 | Student clicks ‘generate schedule button’ with a set of time preferences and for given years, and clicks ‘validate schedule’ without choosing any tutorial and laboratory sections. | | Error message indicating to select tutorial and laboratory section pops up. | Empty error message pops up:  [] | Fail |
| 6.7 | Student choose a day, a set time preferences with the end time before the start time. | | No courses are displayed: ‘no result’ | No courses are displayed: ‘no result’ | **Pass** |

View Saved Schedules

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC7** | | **View Saved Schedules** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 7.1 | Student chooses to view saved schedules | | List of saved schedules is displayed | List of saved schedules is displayed | **Pass** |
| 11.2 | Student chooses to view a specific saved schedule | | Saved schedule is displayed | Saved schedule is displayed | **Pass** |
| 14.3 | Student chooses to delete a saved schedule | | A prompt to confirm deletion is given, then after confirmation the saved schedule is removed | A prompt to confirm deletion is given, then after confirmation the saved schedule is removed | **Pass** |

View Academic Record

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC8** | | **View Academic Record** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 8.1 | Student requests to view Academic Record | | Page with a record of classes passed and the student’s gpa is displayed | Page with a record of classes is displayed, but no classes on the record | **Pass** (with errors when no classes completed) |

Drop Course

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC9** | | **Drop Course** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 9.1 | Student requests to drop a non-prerequisite course, and confirms the choice | | Updated schedule is produced and displayed | Updated schedule is produced and displayed | **Pass** |
| 9.2 | Student requests to drop a non-prerequisite course, and cancels the confirmation dialog box | | Same schedule is displayed | Same schedule is displayed | **Pass** |
| 9.3 | Student requests to drop a course that is a prerequisite to another course saved later in the schedule, and confirms the choice | | Message indicating that that course could not be dropped due to it being a prerequisite for a later course that’s saved is displayed | The course is successfully dropped from the schedule, and the course requiring that prerequisite stays in the schedule | Fail |

Add Course

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~UC10~~** | | **~~Add Course~~** | | | |
| ~~Test ID~~ | ~~Description~~ | | ~~Expected Output~~ | ~~Actual Output~~ | ~~Result~~ |
| ~~10.1~~ | ~~Student requests to view list of saved schedules~~ | | ~~List of saved schedules is displayed~~ | ~~List of saved schedules is displayed~~ | **~~Pass~~** |
| ~~10.2~~ | ~~Student requests to view a saved schedule~~ | | ~~A schedule is displayed~~ | ~~A schedule is displayed~~ | **~~Pass~~** |
| ~~10.3~~ | ~~Student requests to add a course to the saved schedule~~ | | ~~A search screen is displayed prompting the user to enter the name of the course~~ | *~~Not Yet Implemented~~* | *~~Not Yet Implemented~~* |

Saved Generated Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC11** | | **Save Generated Schedules** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 11.1 | Student chooses to save a generated schedule | | Saved schedule confirmation is displayed | Saved schedule confirmation is displayed | **Pass** |
| 11.2 | Student chooses to save a schedule in a semester that already has a saved schedule | | Error message indicating the semester already has a schedule saved | Saved schedule confirmation is displayed | Fail |

View Weekly Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~UC12~~** | | **~~View Weekly Schedule~~** | | | |
| ~~Test ID~~ | ~~Description~~ | | ~~Expected Output~~ | ~~Actual Output~~ | ~~Result~~ |
| ~~12.1~~ | ~~Student requests to see a saved schedule.~~ | | ~~Displays a prompt message to let student select a week.~~ | ~~Not implemented~~ | ~~Not implemented~~ |
| ~~12.2~~ | ~~From the prompted message of week selection, student selects a week.~~ | | ~~Displays the student’s schedule for the selected week.~~ | ~~Not implemented~~ | ~~Not implemented~~ |

Manage Courses (Administrator)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC13** | | **Manage Courses** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 13.1 | Administrator requests to manage courses | | Course management page is displayed | Course management page is displayed | **Pass** |
| 13.2 | Administrator enters an existing course I.D. to the search bar | | Course with the given I.D. is displayed, along with an option to edit the course | Course with the given I.D. is displayed, along with the course code, course description, and an option to view, edit, or delete the course | **Pass** |
| 13.3 | Administrator enters a non-existent course I.D. to the search bar | | Message indicating that a course with that I.D. does not exist | “No results found” message is displayed | **Pass** |
| 13.4 | Administrator enters an existent course code to the search bar | | Course with the given code is displayed, along with an option to edit the course | Course with the given code is displayed, along with the course I.D., course description, and an option to view, edit, or delete the course | **Pass** |
| 13.5 | Administrator enters a non-existent course code to the search bar | | Message indicating that a course with that course code does not exist | “No results found” message is displayed | **Pass** |
| 13.6 | Administrator enters an existent course description to the search bar | | Course with the given description is displayed, along with an option to edit the course | Course with the given description is displayed, along with the course I.D., course code, and an option to view, edit, or delete the course | **Pass** |
| 13.7 | Administrator enters a non-existent course description to the search bar | | Message indicating that a course with that description does not exist | “No results found” message is displayed | **Pass** |
| 13.8 | Administrator requests to delete a course and confirms dialog box | | Updated course bank without the deleted course is produced and displayed | Updated course bank without the deleted course is produced and displayed | **Pass** |
| 13.9 | Administrator requests to delete a course and cancels dialog box | | Same course bank including the chosen course is displayed with no changes | Same screen as before requesting course deletion is displayed | **Pass** |
| 13.10 | Administrator requests to update a course | | Course editing page is displayed | Course editing page is displayed | **Pass** |
| 13.11 | Administrator edits the course code of a course | | Updated course bank with the new code is produced and displayed | Updated course details with the new course code is displayed | **Pass** |
| 13.11 | Administrator edits the course code of a course | | Updated course bank with the new description is produced and displayed | Updated course details with the new course description is displayed | **Pass** |
| 13.12 | Administrator edits the number of credits of a course | | Updated course bank with the new number of credits is produced and displayed | Updated course details is displayed, and the new number of credits is produced | **Pass** |
| 13.13 | Administrator edits the type of the course | | Updated course bank with the new course type is produced and displayed | Updated course details is displayed, and the new course type is produced | **Pass** |

Create Course

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC14** | | **Create Course** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 14.1 | Administrator requests to create a course | | Course creation page is displayed | Create Course page is displayed | **Pass** |
| 14.2 | Administrator enters a course ID, description and amount of credits | | Course info is registered and displayed | Course info is registered and displayed | **Pass** |
| 14.3 | Administrator enters a description and credits, but no course ID | | Error message indicating the course ID field has been left blank | Please fix the following input errors:   * Course Code cannot be blank. | **Pass** |
| 14.4 | Administrator enters a course ID and credits, but no description | | Error message indicating the description field has been left blank | Please fix the following input errors:   * Course Description cannot be blank. | **Pass** |
| 14.5 | Administrator enters a course ID and description, but no credits | | Error message indicating the credits field has been left blank | Please fix the following input errors:   * Credits cannot be blank. | **Pass** |

View Courses (Administrator)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC15** | | **View Courses** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 15.1 | Administrator requests to view courses | | List of all courses in the system is displayed | List of all courses in the system is displayed | **Pass** |
| 15.2 | Administrator clicks the create course link | | Create course page is displayed | Create course page is displayed | **Pass** |
| 15.3 | Administrator clicks the manage course link | | Manage courses page is displayed | Manage courses page is displayed | **Pass** |
| 15.4 | Administrator clicks on a course prerequisite | | Course information for that course is displayed | Course information for that course is displayed | **Pass** |

Add User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC16** | | **Add User** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 16.1 | Administrator requests to add a user | | Create user page is displayed | Create user page is displayed | **Pass** |
| 16.2 | Administrator clicks the create button | | Error message indicating that all fields have been left blank | Error: please fix the following input errors   * Username cannot be blank * Firstname cannot be blank * Lastname cannot be blank * Net Name cannot be blank * Password cannot be blank | **Pass** |
| 16.3 | Administrator enters a firstname, lastname, netname, password, but no username. | | Error message indicating the Username field has been left blank | Error: please fix the following input errors   * Username cannot be blank | **Pass** |
| 16.4 | Administrator enters a username, lastname, netname, password, but no firstname. | | Error message indicating the Firstname field has been left blank | Error: please fix the following input errors   * Firstname cannot be blank | **Pass** |
| 16.5 | Administrator enters a username, firstname, netname, password, but no lastname. | | Error message indicating the Lastname field has been left blank | Error: please fix the following input errors   * Lastname cannot be blank | **Pass** |
| 16.6 | Administrator enters a username, firstname, lastname, password, but no netname. | | Error message indicating the Net Name field has been left blank | Error: please fix the following input errors   * Net Name cannot be blank | **Pass** |
| 16.7 | Administrator enters a username, firstname, lastname, netname, but no password. | | Error message indicating the Password field has been left blank | Error: please fix the following input errors   * Password cannot be blank | **Pass** |
| 16.8 | Administrator enters a password of less than 6 characters | | Error message indicating the Password is too short | Error: please fix the following input errors   * Password is too short | **Pass** |
| 16.9 | Administrator enters a username, firstname, lastname, netname, and a password. | | User info is registered and displayed | User info is registered and displayed | **Pass** |

Search Users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC17** | | **Search for Users** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 17.1 | Administrator leaves the search field blank and press ‘Search’ button. | | Displays all users without any filter. | Displays all users without any filter. | **Pass** |
| 17.2 | Administrator enters the ID in the ID search field of an inexistent user. | | No user found. | No user found. | **Pass** |
| 17.3 | Administrator enters the ID in the ID search field of an existent user. | | Displays a row corresponding to the user. | Displays a row corresponding to the user. | **Pass** |
| 17.4 | Administrator uses comparative operators in the ID search field. | | Displays the users according to the comparative operators. | Displays the users according to the comparative operators. | **Pass** |
| 17.5 | Administrator enters a text in the username search field which is not found in any usernames. | | No user found. | No user found. | **Pass** |
| 17.6 | Administrator enters an existing username in the username search field. | | Displays a row corresponding to the user with that username. | Displays a row corresponding to the user with that username. | **Pass** |
| 17.7 | Administrator enters a portion of an existing username in the username search field. | | Displays one or many rows which username contains the text entered. | Displays one or many rows which username contains the text entered. | **Pass** |
| 17.8 | Administrator enters an existent username in the username search field without respecting upper/lower cases of the actual username. | | Displays a row corresponding to the user with that username. | Displays a row corresponding to the user with that username. | **Pass** |
| 17.9 | Administrator enters a text in the first name search field which is not found in any first names. | | No user found. | No user found. | **Pass** |
| 17.10 | Administrator enters an existing first name in the appropriate search field. | | Displays one or many rows which user’s first name contains the text entered as search criteria. | Displays one or many rows which user’s first name contains the text entered as search criteria. | **Pass** |
| 17.11 | Administrator enters a portion of an existing first name in the appropriate search field. | | Displays one or many rows which user’s first name contains the text entered as search criteria. | Displays one or many rows which user’s first name contains the text entered as search criteria. | **Pass** |
| 17.12 | Administrator enters an existent first name in the appropriate search field without respecting upper/lower cases of the actual first name. | | Displays one or many rows which user’s first name contains the text entered as search criteria. | Displays one or many rows which user’s first name contains the text entered as search criteria. | **Pass** |
| 17.13 | Administrator enters a text in the last name search field which is not found in any last names. | | No user found. | No user found. | **Pass** |
| 17.14 | Administrator enters an existing last name in the appropriate search field. | | Displays one or many rows which user’s last name contains the text entered as search criteria. | Displays one or many rows which user’s last name contains the text entered as search criteria. | **Pass** |
| 17.15 | Administrator enters a portion of an existing last name in the appropriate search field. | | Displays one or many rows which user’s last name contains the text entered as search criteria. | Displays one or many rows which user’s last name contains the text entered as search criteria. | **Pass** |
| 17.16 | Administrator enters an existent last name in the appropriate search field without respecting upper/lower cases of the actual first name. | | Displays one or many rows which user’s last name contains the text entered as search criteria. | Displays one or many rows which user’s last name contains the text entered as search criteria. | **Pass** |
| 17.17 | Administrator enters a text in the net name search field which is not found in any last names. | | No user found. | No user found. | **Pass** |
| 17.18 | Administrator enters an existing net name in the appropriate search field. | | Displays one or many rows which user’s net name contains the text entered as search criteria. | Displays one or many rows which user’s net name contains the text entered as search criteria. | **Pass** |
| 17.19 | Administrator enters a portion of an existing net name in the appropriate search field. | | Displays one or many rows which user’s net name contains the text entered as search criteria. | Displays one or many rows which user’s net name contains the text entered as search criteria. | **Pass** |
| 17.20 | Administrator enters an existent net name in the appropriate search field without respecting upper/lower cases of the actual first name. | | Displays one or many rows which user’s net name contains the text entered as search criteria. | Displays one or many rows which user’s net name contains the text entered as search criteria. | **Pass** |

View User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC18** | | **View User** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 18.1 | Administrator clicks the magnifying glass button from a user’s row. | | Displays the profile of the corresponding user with administrative operations. | Displays the profile of the corresponding user with administrative operations. | **Pass** |

Update User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC19** | | **Update User** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 19.1 | Administrator clicks the pencil button from a user’s row. | | Displays the editable profile of the corresponding user with current data filled in the editable profile. | Displays the editable profile of the corresponding user with current data filled in the editable profile. | **Pass** |
| 19.2 | From a user’s profile viewed by an administrator, the administrator clicks on the ‘update user’ operation. | | Displays the editable profile of the corresponding user with current data filled in the editable profile. | Displays the editable profile of the corresponding user with current data filled in the editable profile. | **Pass** |
| 19.3 | Administrator attempts to save the updated profile with empty username. | | Displays an error saying that the username cannot be blank. | Displays an error saying that the username cannot be blank. | **Pass** |
| 19.4 | Administrator attempts to save the updated profile with empty first name. | | Displays an error saying that the first name cannot be blank. | Displays an error saying that the first name cannot be blank. | **Pass** |
| 19.5 | Administrator attempts to save the updated profile with empty last name. | | Displays an error saying that the last name cannot be blank. | Displays an error saying that the last name cannot be blank. | **Pass** |
| 19.6 | Administrator attempts to save the updated profile with empty net name. | | Displays an error saying that the net name cannot be blank. | Displays an error saying that the net name cannot be blank. | **Pass** |
| 19.7 | Administrator attempts to save the updated profile with empty password. | | Displays an error saying that the password cannot be blank. | Displays an error saying that the password cannot be blank. | **Pass** |
| 19.8 | Administrator attempts to save the updated profile with an already existing username. | | Displays an error saying that the username is already taken. | Displays an error saying that the username is already taken. | **Pass** |
| 19.9 | Administrator attempts to save the updated profile with an already existing net name. | | Displays an error saying that the net name is already taken. | Displays an error saying that the net name is already taken. | **Pass** |
| 19.10 | Administrator attempts to save the updated profile with a password which length is lower than six characters. | | Displays an error saying that the password is too short and that the minimum length is six characters. | Displays an error saying that the password is too short and that the minimum length is six characters. | **Pass** |
| 19.11 | Administrator attempts to save the updated profile with any combination of invalidity of test cases 19.3 to 19.10 | | Displays a list of all invalidity to be corrected. | Displays a list of all invalidity to be corrected. | **Pass** |
| 19.12 | Administrator hits the save button without changing any fields. | | Updated profile is displayed, all information including password remains unchanged. | Password has been changed | Fail |
| 19.13 | Administrator attempts to save the updated profile with an inexistent username. | | Updated profile is displayed, all remaining information have remained unchanged. | Updated profile is displayed, all remaining information have remained unchanged. | **Pass** |
| 19.14 | Administrator attempts to save the updated profile with a new first name. | | Updated profile is displayed, all remaining information have remained unchanged. | Updated profile is displayed, all remaining information have remained unchanged. | **Pass** |
| 19.15 | Administrator attempts to save the updated profile with a new last name. | | Updated profile is displayed, all remaining information have remained unchanged. | Updated profile is displayed, all remaining information have remained unchanged. | **Pass** |
| 19.16 | Administrator attempts to save the updated profile with an inexistent net name. | | Updated profile is displayed, all remaining information have remained unchanged. | Updated profile is displayed, all remaining information have remained unchanged. | **Pass** |
| 19.17 | Administrator attempts to save the updated profile with a new password which length is greater than six. | | Updated profile is displayed, password for that user has been changed. | Updated profile is displayed, password for that user has been changed. | **Pass** |
| 19.18 | Administrator changes the privilege of the user from student to administrator. | | Updated profile is displayed, user is now administrator. | Updated profile is displayed, user is now administrator. | **Pass** |
| 19.19 | Administrator clicks on the password field but does not change anything, save the profile. | | Update profile is displayed, user information and password have remained unchanged. | Password has been changed. | Fail |

Delete User

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC20** | | **Delete User** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 20.1 | Administrator hit the ‘X’ button on a row associated to a user in a table of users. | | Displays a confirmation message to delete the corresponding user. | Displays a confirmation message to delete the corresponding user. | **Pass** |
| 20.2 | From the profile page of a user viewed as an administrator, the administrator clicks the ‘Delete User’ operation. | | Displays a confirmation message to delete the corresponding user. | Displays a confirmation message to delete the corresponding user. | **Pass** |
| 20.3 | The administrator hits the cancel button on the confirmation message to delete a user. | | Confirmation message disappears, user remains on the system. | Confirmation message disappears, user remains on the system. | **Pass** |
| 20.4 | The administrator hits the ‘OK’ button on the confirmation message to delete a user. | | The corresponding user is removed from the system, the administrator is brought back to the user management page with an updated table of users. | The corresponding user is removed from the system, the administrator is brought back to the user management page with an updated table of users. | **Pass** |

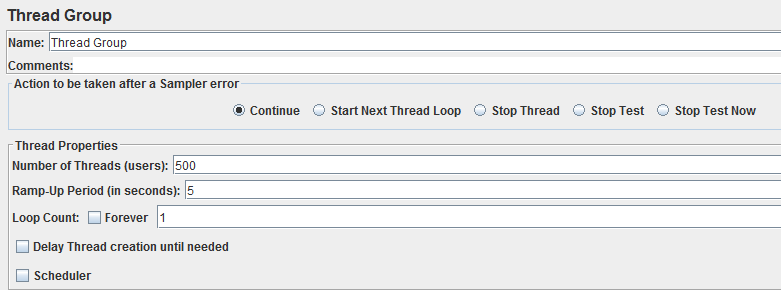
Browse Users

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **UC21** | | **Browse Users** | | | |
| Test ID | Description | | Expected Output | Actual Output | Result |
| 18.1 | Admin selects user’s tab and is populated with all the users | | List of users are displayed on screen | List of users are displayed on screen | Pass |

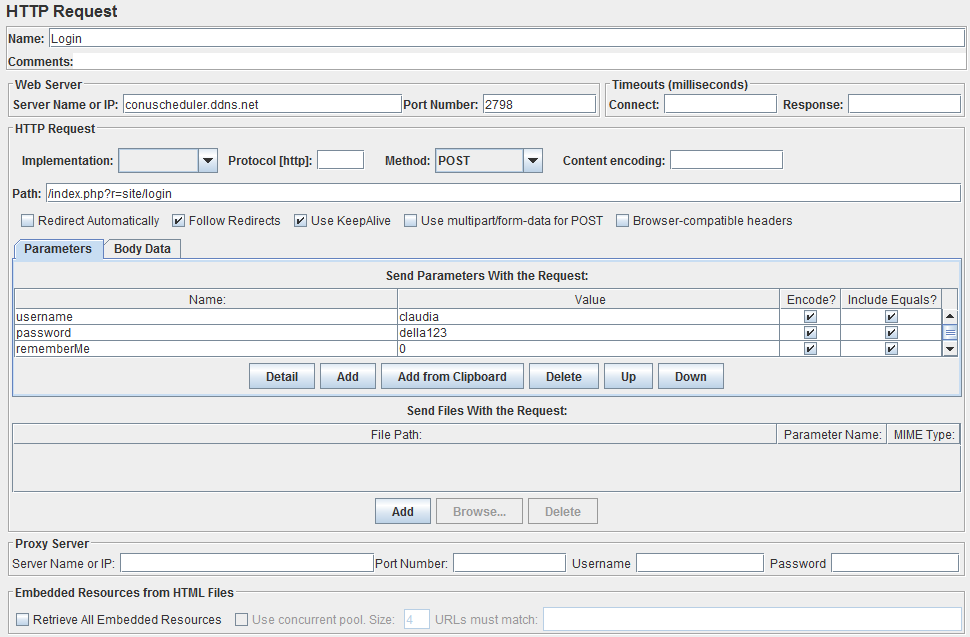
3.3.3 Stress Testing

We ran the JMeter tool with 500 simultaneous users and a 5-second ramp-up period (time to get all threads started). Each thread sent an HTTP request to login to the system and then another request to generate a schedule.

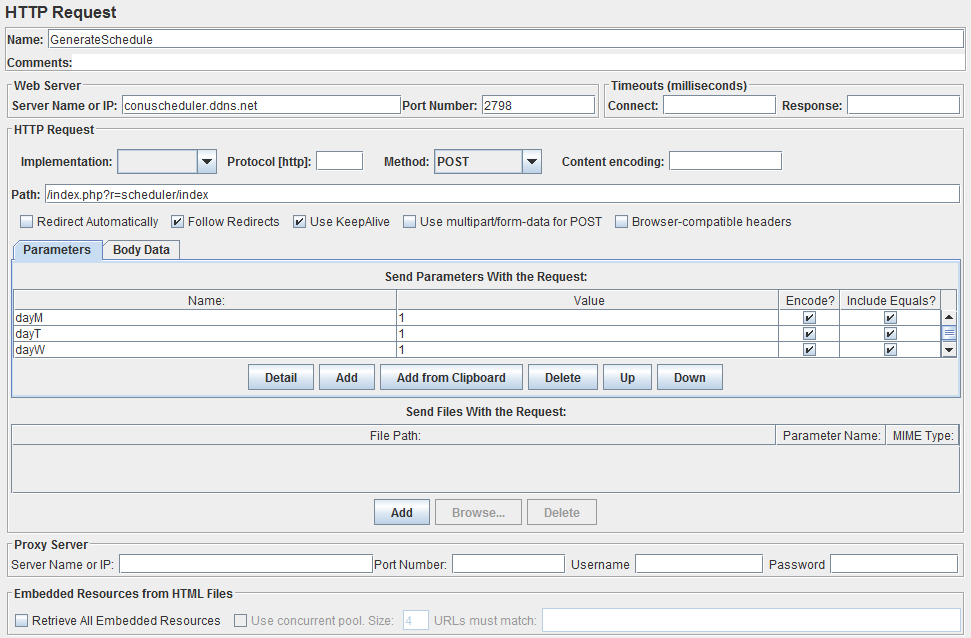
Thread group setup:



Login request setup:

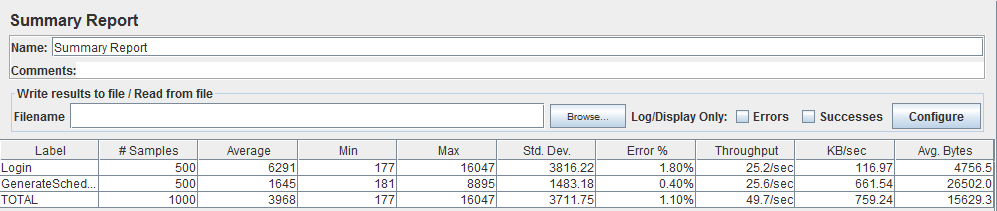


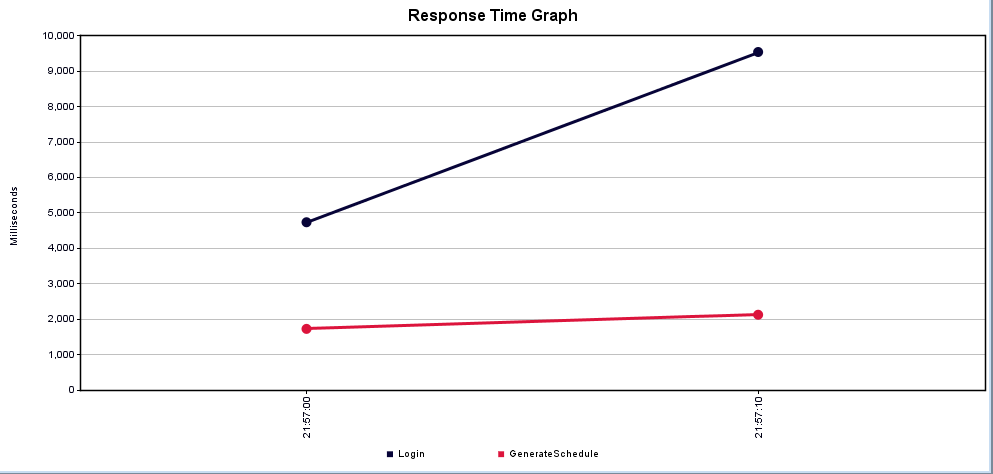
Schedule generation request setup:



Note: the *Parameters* table could not be expanded for the screenshot. The parameters were set up such that the all checkboxes of the preference form were checked off, all start times were set to ‘8:45:00’, and all end times were set to ’23:30:00’, in order to generate a schedule with as many options as possible and therefore stress the system.

The test resulted in zero errors (response time out, system crash, etc), indicating that it can handle many simultaneous users without major problems. We obtained the following summary report and graph of response time:



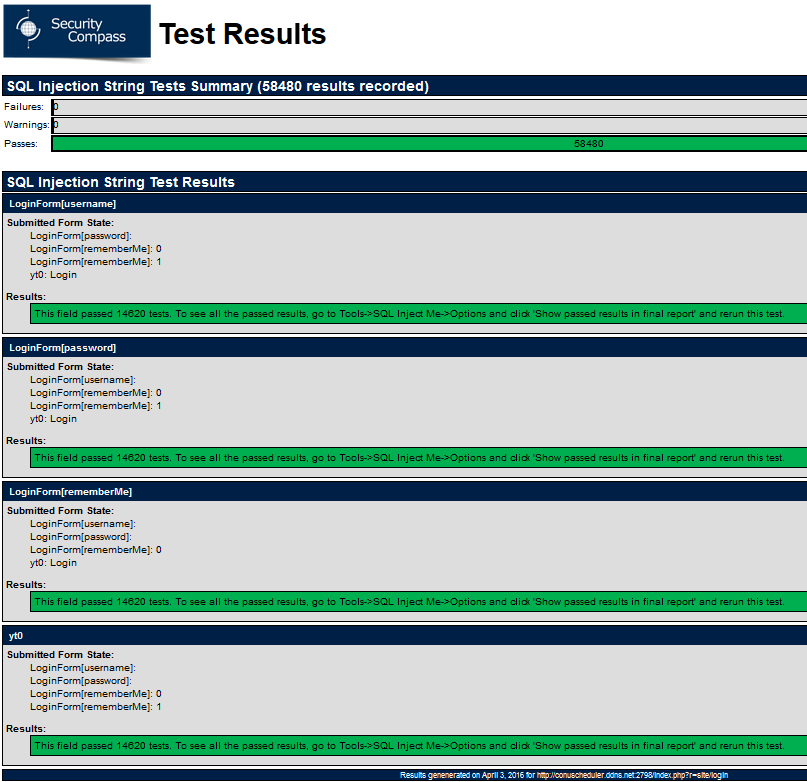


From the above graph, we can see that the response time to login to the system increases as more users try to access it at the same time. However, once logged in, the time to generate a schedule remains fast even when many users are making simultaneous requests.

Security Testing

SQL Inject Me LoginForm

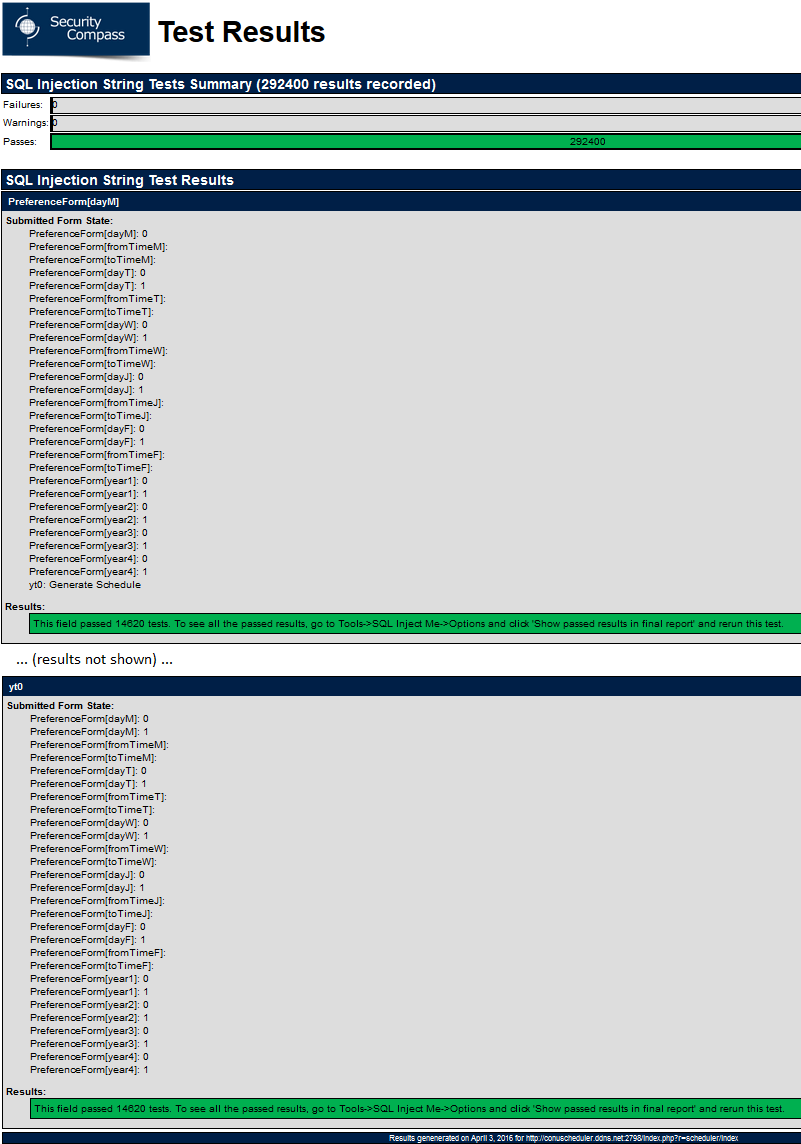
The LoginForm passed all 58480 tests, as depicted in the figure below.

**

*Figure 3.2.2 SQL Inject Me tests on LoginForm*

SQL Inject Me PreferenceForm

The PreferenceForm passed all 292400 tests, as depicted in Figure 3.2.3.



*Figure 3.2.3 SQL Inject Me tests on PreferenceForm*

Nikto results:

