Time Sheet Web App
COMP3910 Assignment 1

Tony Pacheco & Danny Di Iorio
October 27, 2018

Table of Contents

Purpose

The Timesheet application will be an online system which allows users to login and access/edit workplace timesheets and allows an administrator to manage the user's accounts.

Scope

The Timesheet application will be implemented without a database and will instead only store data as long as the server is left running. Because of this, it will also not be a truly distributed application; only keeping data on the local machine running it.

Perspective

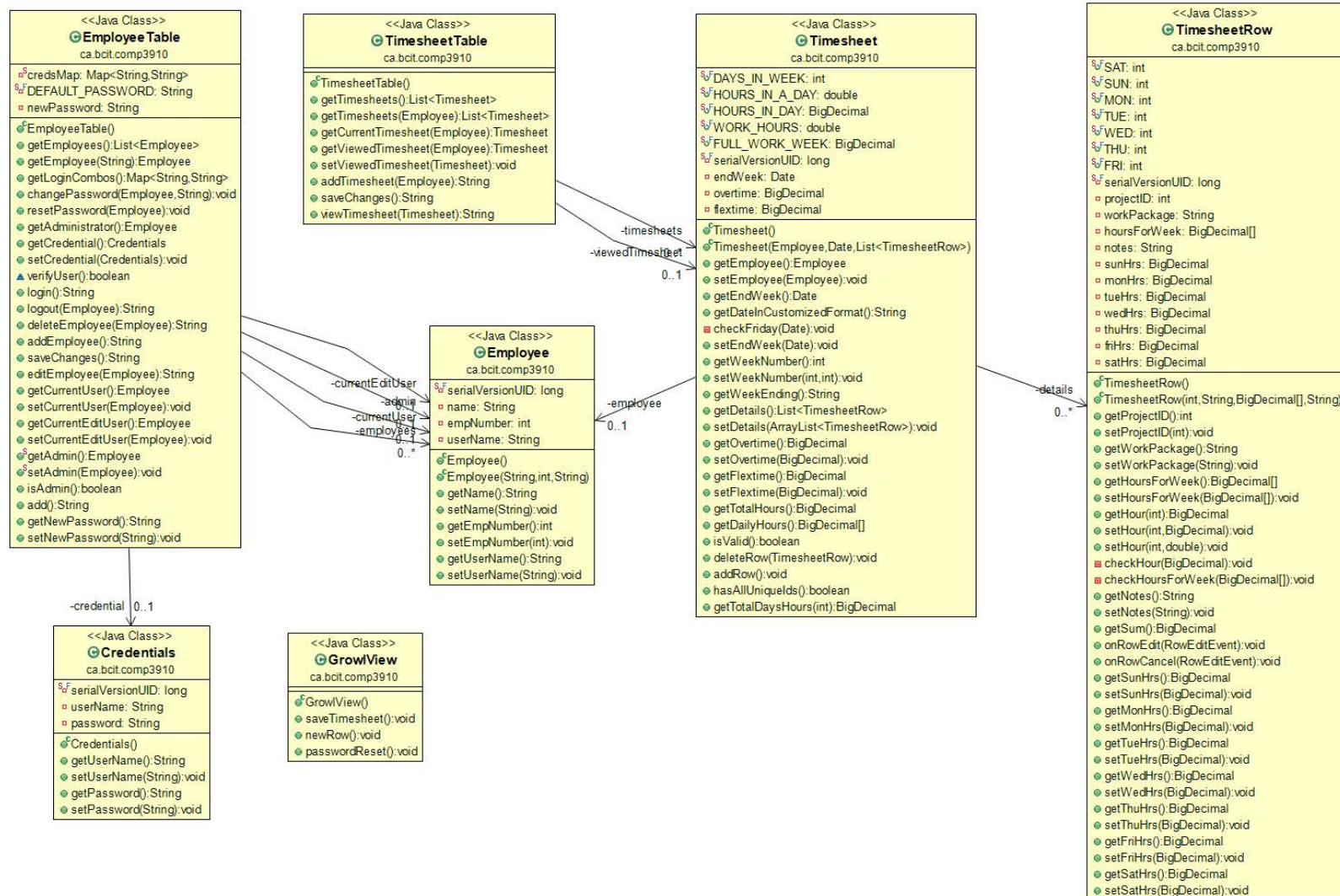The application will store the following data in its Java backend:

1. User (Employee) Data for each user in the system
   a. Name
   b. Employee Number
   c. Username
   d. Password

2. Timesheet Data
   Each timesheet records the hours worked by a single employee on a given week. Each will contain the following data:
   a. Employee Number
   b. Employee Name
   c. Week Number (0-52)
   d. Week Specifier (the date on which the week ends)
   e. The following, each as a set of data to displayed as the rows of a timesheet table
      a. Project Number
      b. Work Package identifier
      c. Total number of hours worked for the week
      d. A column for each day of the week containing the number of hours worked that day
   e. Additional optional notes
   Each row above represents a week of work hours on a given work package of a specific project.

Class Diagram

EmployeeTable
<<Java Class>>
ca.bcit.comp3910

- credsMap: Map<String,String>
- DEFAULT_PASSWORD: String
- newPassword: String

- EmployeeTable()
- getEmployees():List<Employee>
- getEmployee(String):Employee
- getLoginCombos():Map<String,String>
- changePassword(Employee,String):void
- resetPassword(Employee):void
- getAdministrator():Employee
- getCredential():Credentials
- setCredential(Credentials):void
- verifyUser():boolean
- login():String
- logout(Employee):String
- deleteEmployee(Employee):String
- addEmployee():String
- saveChanges():String
- editEmployee(Employee):String
- getCurrentUser():Employee
- setCurrentUser(Employee):void
- getCurrentEditUser():Employee
- setCurrentEditUser(Employee):void
- getAdmin():Employee
- setAdmin(Employee):void
- isAdmin():boolean
- add():String
- getNewPassword():String
- setNewPassword(String):void

TimesheetTable
<<Java Class>>
ca.bcit.comp3910

- TimesheetTable()
- getTimesheets():List<Timesheet>
- getTimesheets(Employee):List<Timesheet>
- getCurrentTimesheet(Employee):Timesheet
- getViewedTimesheet(Employee):Timesheet
- setViewedTimesheet(Timesheet):void
- addTimesheet(Employee):String
- saveChanges():String
- viewTimesheet(Timesheet):String

Timesheet
<<Java Class>>
ca.bcit.comp3910

- DAYS_IN_WEEK: int
- HOURS_IN_A_DAY: double
- HOURS_IN_DAY: BigDecimal
- WORK_HOURS: double
- FULL_WORK_WEEK: BigDecimal
- serialVersionUID: long
- endWeek: Date
- overtime: BigDecimal
- flextime: BigDecimal

- Timesheet()
- Timesheet(Employee,Date,List<TimesheetRow>)
- getEmployee():Employee
- setEmployee(Employee):void
- getEndWeek():Date
- getDateInCustomizedFormat():String
- checkFriday(Date):void
- setEndWeek(Date):void
- getWeekNumber():int
- setWeekNumber(int,int):void
- getWeekEnding():String
- getDetails():List<TimesheetRow>
- setDetails(ArrayList<TimesheetRow>):void
- getOvertime():BigDecimal
- setOvertime(BigDecimal):void
- getFlextime():BigDecimal
- setFlextime(BigDecimal):void
- getTotalHours():BigDecimal
- getDailyHours():BigDecimal[]
- isValid():boolean
- deleteRow(TimesheetRow):void
- addRow():void
- hasAllUniqueIds():boolean
- getTotalDaysHours(int):BigDecimal

TimesheetRow
<<Java Class>>
ca.bcit.comp3910

- SAT: int
- SUN: int
- MON: int
- TUE: int
- WED: int
- THU: int
- FRI: int
- serialVersionUID: long
- projectID: int
- workPackage: String
- hoursForWeek: BigDecimal[]
- notes: String
- sunHrs: BigDecimal
- monHrs: BigDecimal
- tueHrs: BigDecimal
- wedHrs: BigDecimal
- thuHrs: BigDecimal
- friHrs: BigDecimal
- satHrs: BigDecimal

- TimesheetRow()
- TimesheetRow(int,String,BigDecimal[],String)
- getProjectID():int
- setProjectID(int):void
- getWorkPackage():String
- setWorkPackage(String):void
- getHoursForWeek():BigDecimal[]
- setHoursForWeek(BigDecimal[]):void
- getHour(int):BigDecimal
- setHour(int,BigDecimal):void
- setHour(int,double):void
- checkHour(BigDecimal):void
- checkHoursForWeek(BigDecimal[]):void
- getNotes():String
- setNotes(String):void
- getSum():BigDecimal
- onRowEdit(RowEditEvent):void
- onRowCancel(RowEditEvent):void
- getSunHrs():BigDecimal
- setSunHrs(BigDecimal):void
- getMonHrs():BigDecimal
- setMonHrs(BigDecimal):void
- getTueHrs():BigDecimal
- setTueHrs(BigDecimal):void
- getWedHrs():BigDecimal
- setWedHrs(BigDecimal):void
- getThuHrs():BigDecimal
- setThuHrs(BigDecimal):void
- getFriHrs():BigDecimal
- setFriHrs(BigDecimal):void
- getSatHrs():BigDecimal
- setSatHrs(BigDecimal):void

Employee
<<Java Class>>
ca.bcit.comp3910

- serialVersionUID: long
- name: String
- empNumber: int
- userName: String

- Employee()
- Employee(String,int,String)
- getName():String
- setName(String):void
- getEmpNumber():int
- setEmpNumber(int):void
- getUserName():String
- setUserName(String):void

Credentials
<<Java Class>>
ca.bcit.comp3910

- serialVersionUID: long
- userName: String
- password: String

- Credentials()
- getUserName():String
- setUserName(String):void
- getPassword():String
- setPassword(String):void

GrowlView
<<Java Class>>
ca.bcit.comp3910

- GrowlView()
- saveTimesheet():void
- newRow():void
- passwordReset():void

-timesheets
-viewedTimesheet
0..1

-currentEditUser
-admin
-currentUser
-employees
0..1
0..*

-employee
0..1

-details
0..*

-credential 0..1

Use Cases

The two types of users which the application will support are general users and the system's administrator. General users will be able to manage their own account settings and create and edit their own timesheets. Administrators will be able to do anything that general users can do, as well as manage the creation, modification, and deletion of other user accounts. Administrators can also view timesheets from all users in the system.
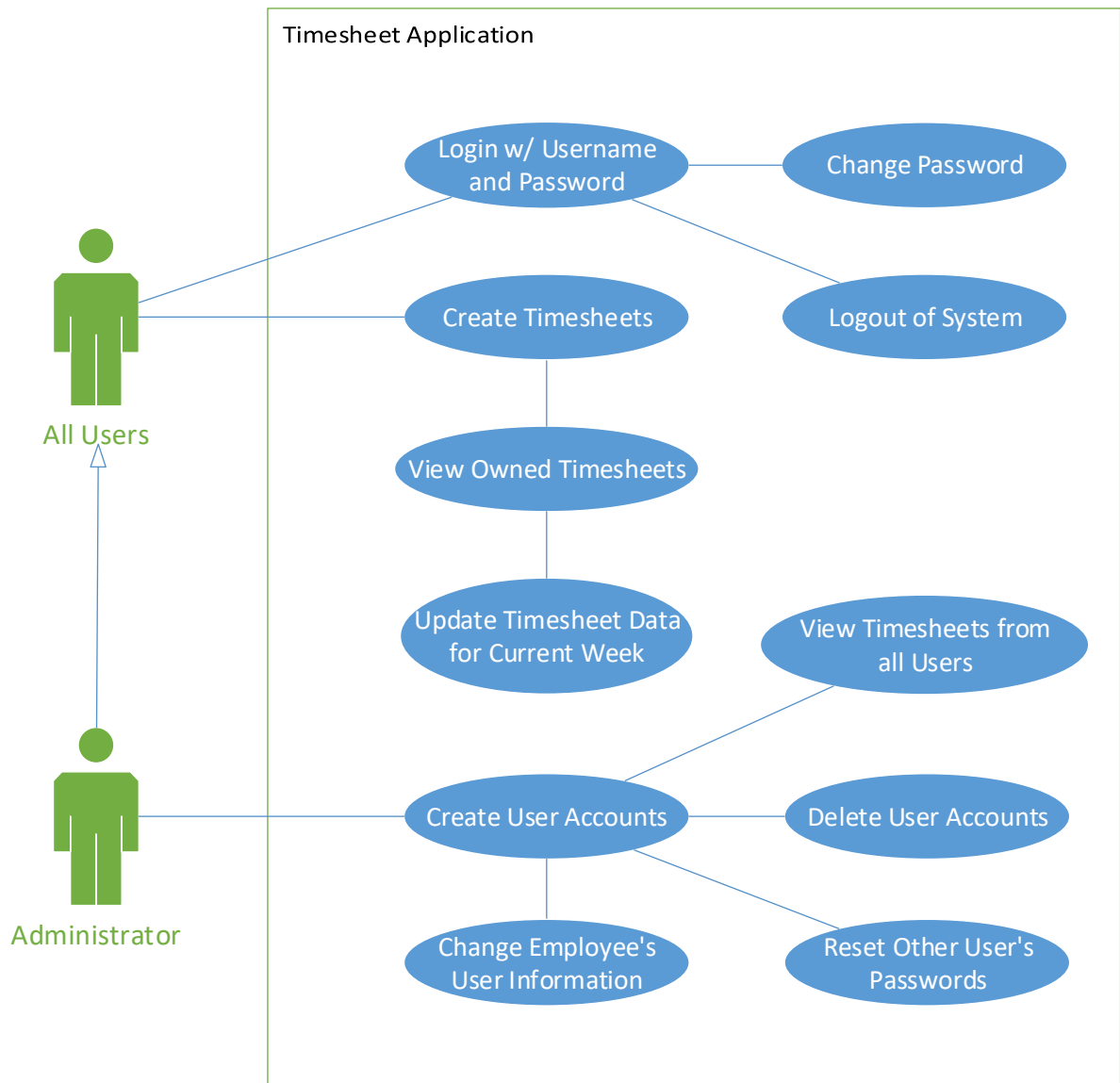
Interface Requirements

General Users

When a user first starts the application, they will see a login page, if they have a registered account, they can login with their username and password. If they do not have a registered account, an administrator will need to create an account for them.

After a registered user successfully logs in, they will see a page with the list of their saved timesheets, organized by date. Each timesheet can be opened and edited by clicking on it.

Users can create a new timesheet by clicking on "New Timesheet" in the header. This will open a page with a new timesheet with five empty rows. By clicking on a pencil icon in the last column of the table, the user will be able to edit the contents of that row in the table, which allows them to input/edit hours worked for any day of the week.

This page will also have buttons on a top toolbar which allow them to view a different week's timesheet, create a new timesheet, change their password, or logout of the application.

Administrators

If an administrator logs in to the application, they will see a page with the list of saved timesheets from all users, organized by date. Each timesheet can be opened and edited by clicking on it.

Admins have a header link to see a page with a list of all the users in the system. They will be able to click on an edit button beside each user to edit them, click a button beside each user to remove them, or click a button at the bottom of the list to create a new user. In the edit user page, the admin can edit all fields and have the option to reset their password to default.

The administrators will also have the same options in the toolbar as the general users so that they can create and edit their own timesheets. They will have access to view other users' timesheets.
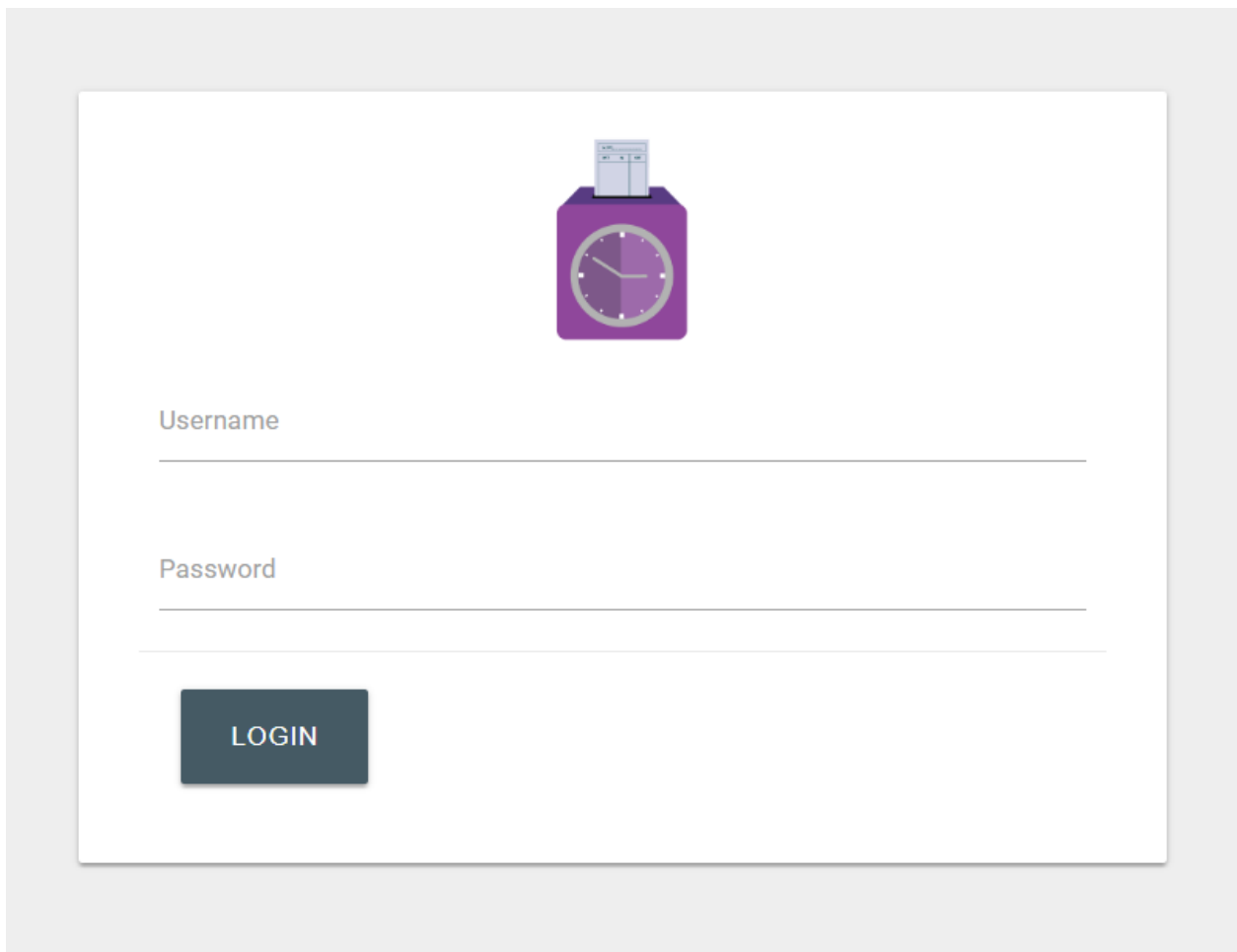
Operating Environment

The back-end will be implemented using Java instead of a database, and a Wildfly 13 server to serve the application webpages.

The front-end will use Java Server Faces and Prime Faces for the user interface. Front-end styling is handled by a custom stylesheet and Materialize 3rd party CSS library. Templates will be used for the main layout, header, and footer, and a message bundle for all text where possible.

The management of the state of the application, and communication between the mock-database and the user interface will be implemented using Java Beans.

UI Screenshots

Login page

View Timesheets page (admin)

New Timesheet page (admin)

Timesheet

Timesheet Details   ⊟

| Employee Number : | 1 | Week Number : | 43 |
| Name : | Tony Pacheco | Week Ending : | 2018-10-26 |

Project	WP	Total	Sat	Sun	Mon	Tue	Wed	Thu	Fri	Notes	
22	WEC	20			5	5		5	5	Complete	✎
11	ABA	14			2	2	8	2		WIP	✎
0		0									✎
4	DDD	13.5	5	5					3.5	Complete next week	✎
0		0									✎
	Total	47.5	5	5	7	7	8	7	8.5		

NEW ROW

SAVE TIMESHEET

[Danny Di Iorio] [Tony Pacheco] [Timesheet Assignment 1]

View users page (admin)

Users

Current Employee User Accounts				
Edit User	**E Number**	**Name**	**Username**	**Remove User**
EDIT	1	Tony Pacheco	tp1	REMOVE
EDIT	2	Danny Di Iorio	dd2	REMOVE
EDIT	3	King Henry	hh3	REMOVE

NEW USER

Edit user page (admin) – new user page is the same except no Password field or Reset Password button

New user page (admin)
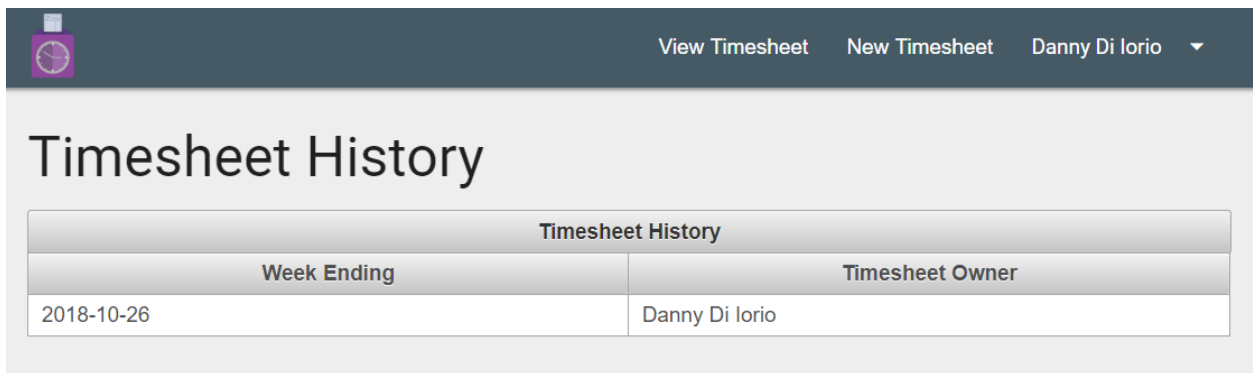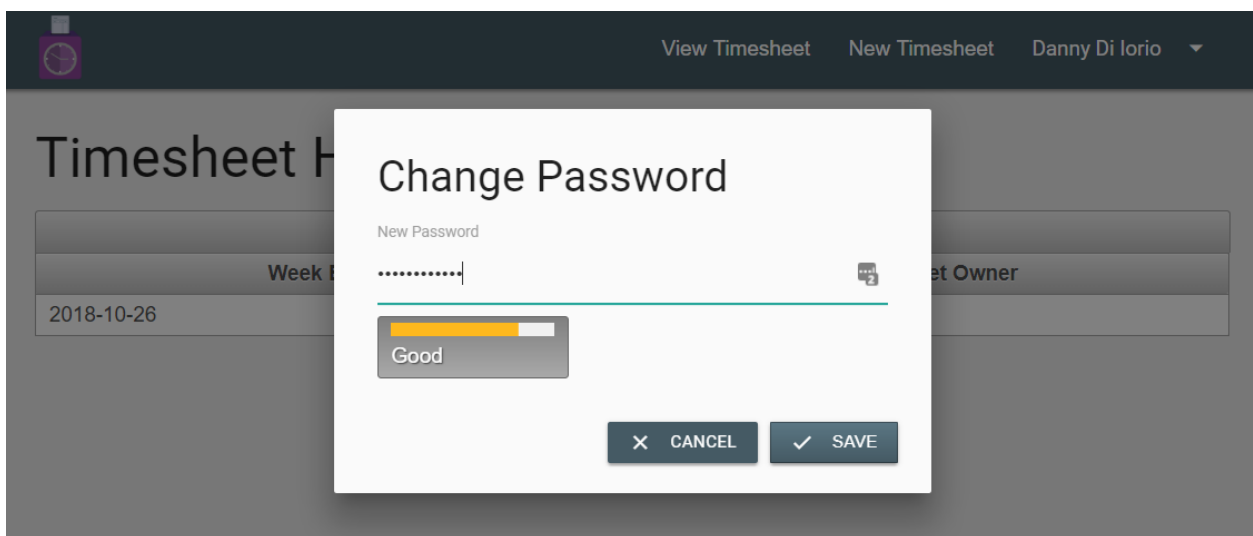
View Timesheets page (regular user)

Change password dialog

Build and Run Instructions

Configurations

- o   Java Development Kit 8
- o   Eclipse Java EE IDE
- o   Wildfly 13 and JBOSS_HOME defined

Importing Project into Eclipse

1) Unzip submitted zip file into your chosen directory
2) In Eclipse, select "File -> Import…"
3) Select "JSF Project", select next
4) Find "web.xml" file from the project directory (WEB-INF folder)
5) Select "Finish"

Build and Deploy Project on Server

1) In Eclipse, stop WildFly server and delete any previously deployed versions of this project
2) Start the WildFly server
3) Right click on the project folder in Eclipse
4) Select "Run As -> Run on Server"
5) Open a web browser and enter "http://localhost:8080/assignment1/"

User Accounts

Admin:
- o Username: tp1
- o Password: pass

User:
- o Username: dd2
- o Password: pass

Test Plan Results

1) Login page – **ALL PASSED**
   a) Users will enter a correct combination of password and username
   b) Validation checks these fields and displays error messages appropriately
2) Logout – **ALL PASSED**
   a) User can logout of their account by pressing the logout button in the header dropdown menu
   b) After pressing the logout button, they are directed to the login page
3) User landing page – **ALL PASSED**
   a) Page with a list of saved timesheets (empty at first)
   b) Header containing links to View Timesheets, New Timesheet, User dropdown menu
4) Administrator landing page – **ALL PASSED**
   a) Same page as users
   b) Header will have an additional link Users, which will lead to a list of users
5) View timesheet – **ALL PASSED**
   a) This page shows a list of saved timesheets, organized by date
   b) The timesheet owner is shown
6) Create timesheet – **ALL PASSED**
   a) When a new timesheet is created five rows will
   b) User details are listed above the timesheet
   c) It will be defaulted as current week
   d) Clicking on the add row button will add a row to the timesheet table
   e) Multiple rows can be edited at once; clicking check mark or X saves or cancels respectively

f) Time entered during the week must be in the unit of hour from 0.0 to 24.0, it may be integer or with one decimal place. Error message shows if validation fails

g) WP (Working Project) must have alphabetical value combined with numerical value

h) Total number of hours should appear before the column of the day of the week column

    i) The calculation is done automatically; however, it is only done when the user clicks the save button

7) View users page – **ALL PASSED**

a) This page will show all the users in a list with all their information

b) The admin can click on a button to edit, directing them to an edit page

c) The admin can also click on a remove button to delete the user

d) The admin can click a button to add a new user

8) Edit users page – **ALL PASSED**

a) The admin can edit all fields related to the user

b) The user's password can be reset to a default value ('*1234*') by clicking the reset password button

c) Validation messages show if required fields are empty

9) New user page – **ALL PASSED**

a) This page allows a manager to create a new user by filling out the following fields:

    i) Employee number - required

    ii) Name - required

    iii) Username - required

    iv) Password

b) Validation messages show if required fields are empty

10) Utilize Timesheet Java project supplied to us – **PARTIAL PASS**

a) We were able to use the Timesheet, TimesheetRow, Employee, and Credentials classes provided; however, connecting our project to the supplied timesheet project and extending our Table classes to it brought on issues we could not solve

b) Instead, we added the above classes to our project and added new methods as required