Individual Assignment Specifications

**Team**: VMS

**Iteration**: Iteration 2

# **Special Roles**

* Project Coordinator: Allen Dorris
* Quality Assurance Czar: Katie Bridson
* Video Demo Creators:
  + Alex Ziegenhorn - Video
* Demo-Booth Operator: Alex Ziegenhorn
* Instructor Meeting Leader: Ben Sell

# **Tasks: Alex Ziegenhorn**

* Task 1 combined: controller/alex
  + Description: Update the invoice and timesheet controllers so that they are functional with the rest of the application, and have it automatically generate an invoice form document when timesheets are approved. Additionally, update the invoice and timesheet html to have a more improved, more organized design.
  + How to Evaluate: The task is complete when the application’s functions work properly(e.g. Clicking buttons do as you expect, it renders entries in the database on a page correctly, etc.) and the invoice and timesheet pages have been updated and function properly.
  + Outcome of Task: This is basically Task 1 and 2 combined. 90% of this work was done in the controller/alex branch, with a few commits being carried out inside of the testing branch. Sadly there was no way around this, due to the nature of the relationship between html and controllers. This task was completed successfully according to evaluation standards.
* Task 2: input-validation/alex
  + Description: Add proper validations to the form input. This includes implementing a notification system to inform the user of any errors, such as when an illegal input submission is attempted.
  + How to Evaluate: The task is complete when the validations are in place, and the user is properly notified after making any mistakes.
  + Outcome of Task: Please note, due to the nature of the relationship between html and controllers and input validation, **I did most of the work on this task in the branch controller/alex**. There was also a good amount of work done in testing branch, sadly there was no way around this as the most recent work by other teammates was necessary to implement the input validation. This task was completed successfully according to evaluation standards.
* NEW\*\*: html/alex
  + Description: Update the design of the employee dashboard to be functionable and look nice.
  + How to Evaluate: The task is complete when the pages look presentable.
  + Outcome of Task: This task was completed on controller/alex and testing branches successfully, according to evaluation standards.

# **Tasks: Allen Dorris**

* Task 1: html/allen
  + Description: Modify the design of the dashboard, and add unified css to the site to make it be uniform and to have a nice looking design that is easy to navigate and use.
  + How to Evaluate: The task is complete when it looks nice.
  + Outcome of Task: Design of employee dashboard was redistributed to Alex Ziegenhorn. Design of admin dashboard was redistributed to Katie Bridson. CSS and site design was pushed back to iteration 3.
* Task 2: security/allen
  + Description: Enable Spring security on the application to ensure that its data is secure. This includes verification of user accounts, sessions, enabling permission restrictions(such as so normal viewers cannot view the admin pages, and guests(users without an account) can only access the login page).
  + How to Evaluate: This task is complete when Spring Security is enabled, i.e. users without accounts are redirected to the login page, information displays for the specific user logged in, and so on.
  + Outcome of Task: Much research and work was put into attempting to implement it, but as of yet this task is incomplete, and a lot of the code that exists does not properly function.
* Task 3: notifications/allen
  + Description: Implement a notification system to inform the user of any errors, such as when an illegal input submission is attempted.
  + How to Evaluate: This task is complete when the user is properly notified after making any mistakes.
  + Outcome of Task: Task was redistributed to Alex Ziegenhorn, as it was discovered to be closely intertwined with the input validations task.

# **Tasks: Ben Sell**

* Task 1: generate-paystub/ben
  + Description: Has the site automatically generate a paystub document once a timesheet has been approved.
  + How to Evaluate: The task is complete when a paystub is automatically generated when a timesheet is approved.
  + Outcome of Task: Task was redefined or redistributed.
* NEW\*\*: paystub/ben
  + Description: Create an HTML page for the paystub form document.
  + How to Evaluate: The task is complete when the HTML page is created.
  + Outcome of Task: Little work has been completed. Very very basic HTML code to display plain text was all that was created.

# **Tasks: Joshua Hatcher**

* Task 1: email/josh
  + Description: Set up email notifications for when the employee’s timesheet due date is approaching, and set up a system where an admin can email employees a registration link to make a new account. The employer should also be notified whenever an employee uploads a timesheet.
  + How to Evaluate: The task is complete the email notification functions properly.
  + Outcome of Task: Email system is not fully integrated with our project, and is currently commented out, but code exists in branch.

# **Tasks: Katie Bridson**

* Task 1: sessions/katie
  + Description: Implement sessions to keep track of information for particular users throughout the page.
  + How to Evaluate: The task is complete when the site changes based on the user signed in.
  + Outcome of Task: Task was redistributed, as it was learned to be a part of Spring Security.
* Task 2: paystubs/katie
  + Description: Update the paystub controller so that it is functional with the rest of the application, and update the paystub html and the employees view to have a more improved, more organized design.
  + How to Evaluate: The task is complete when all of the paystub functionality (creating a paystub with correct calculations and timestamp values, adding a check to the paystub, returning the associated timesheet for corrections, updating the status of the paystub, regenerating a paystub after corrections to the timesheet have been made) is complete and there is appropriate html that can be used to interact with the paystub system,
  + Outcome of Task: I ended up changing much more than just the controller and the html to get this working. There were some model changes and a few additions to the repos and services. All the paystub functionality is in, the objects save correctly to the database, and there are appropriate html and forms that the employee/admin can use. This task is complete.
* NEW\*\*: adminFunctionality/katie
  + Description: Update the design and content of the admin dashboard to be functionable. Also, create any functionality for admin not already created.
  + How to Evaluate: The task is complete when the “admin” can use the admin page to navigate all of the admin features of the app as follows: add a vendor, add a project to a vendor, add an employee to a project, create a new employee stub account, edit permission level of an employee (owner), issue new timesheet to employees, approve timesheet, create paystub, add check to paystub, regenerate paystub, create an invoice, regenerate invoice.
  + Outcome of Task: Because there is no login authentication in the app right now, Alex and I hard coded in employee ids for both the admin and employee views so that will need to be changed next iteration. But with the fake login, all the above admin functionality is in and available on the admin page except for creating a stub employee account and editing an employee’s permission level which should only be done by someone of “owner” level. I completed the features of adding a project to a vendor, adding an employee to a project, issuing a new timesheet, creating a paystub, adding a check to the paystub and regenerating the paystub. I also did some bug fixes on the ones that were already created. The task is not quite complete because of the hardcoded id values and the stub account feature will be made by Alex next iteration. Also viewing all employees and editing them should be added to the list on the admin page.