**Web Based Project Manager**

**Introduction**

 Ruby on rails is a strikingly popular web-based framework amongst the developer community. I plan to create a job setup portion of a project manager for building labor companies to improve project organization. I will be creating a Ruby on Rails application using the MVC framework. The project manager will be used for such companies to manage project specific categories and associations. Some of the project specific categories that I will be creating include job setup, contacts, equipment, budgets, people, codes and a few others. With login functionality, the project manager will allow users to manage many projects at a time, along with their categories. The rails MVC will be used to correlate the associations between the models, views and controllers.

 Each category will correspond to a model, which is a table in the database. When a user navigates to each category, a form will be displayed with records belonging to that project. The user will be able to add/reuse (from a global listing), edit or update records. The user can also choose to add a new record that does not yet exist. For example, if a user wanted to add a piece of equipment to a project, they could add a piece that is already stored in the global equipment table, or choose to create a new piece of equipment, where they would be redirected to add a new equipment record to the global table.

 Records in the global tables will be available to add to any project. A global record of any sort could essentially belong to many projects. This is where the concept of associative relationships in the models come into play.

 The relationships between database tables will be handled in the models. For this application, there will be a total of around nine global tables such as projects, equipment, and people, for example which will need to have organized relationships with the project categories.

 To get a brief picture of the UI, the application will include a navigation menu with a button linking to each project setup category. The user will be able to add, edit or just view each category and its listings. Each project category will have a relationship between two global models. For example, if there are global equipment and project tables in the database, the equipment belonging to a project would be stored in an associative, in-between table that keeps track of all equipment belonging to all projects. A collection in the view will be used to iterate through the equipment that belongs to that project.

 The project will also include user logins with privileged roles. For example, the role of administrator will have more privileges as opposed to just a basic user, whom may only have read privileges. Authentication and authorization will be used to create and manage users and their roles.

**Note:** Northern Michigan University will be unable to keep the project code due to other ownership rights, but will be able to view it for presentation purposes.

**Key Concepts**

|  |  |
| --- | --- |
| * Relational Databases/model relationships
* Global tables vs association tables
 | * DOM Traversal/jQuery
* Rails parameters
 |
| * SQL
 | * Ajax
 |
| * The MVC Framework
* Object Oriented Design
 | * Test first programming
 |

**Grading Scale**

|  |  |
| --- | --- |
| **Grade** | **Included features and knowledge** |
| A | * User logins
* User privileges according to role
* Professional UI/ easy for user
* Use of jQuery for autofill/ prediction
* Use of rails gems
* No bugs upon redirect and save
* Relationships between models/tables
* Alert user of wrong/incomplete inputs
* Understanding of the MVC
* Good coding style/naming conventions
 |
| B | * User logins (no roles/privileges)
* Professional UI
* Use of rails gems
* None/few bugs upon redirect and save
* Relationships between models/tables
* Good understanding of the MVC
* Ok coding style/naming conventions
* Code in the view that could be placed into a method
 |
| C | * User logins (no roles/privileges)
* UI is messy, confusing for user
* Incomplete categories/ not related to other tables
* Save works
* Not user friendly
* Ok understanding of the MVC
* Unclear naming conventions/sloppy methods
* Code in the view that could be placed into a method
 |
| D | * No user logins
* UI is messy with unfinished work
* Categories don’t have relations to other tables
* Confusion about the MVC is apparent
 |