Madalyn Gort and Annalee Halbert

Ecocion, Inc. Project Management System

June 17, 2014

Contents

I. Introduction

A. Client Description

Ecocion is a global provider of environmental solutions and services. Through collaboration with clients, Ecocion helps with problems like regulatory compliance, emission tracking, and environmental impact, and has been utilized by Fortune 500's leading companies (*ecocion*). Ecocion was founded about 10 years ago by Greg George and Tom Duong, and has since grown to employ approximately 30 full-time employees. The Software Engineering department of Ecocion is relatively new, as it is only about two years old. It has grown from one employee, our client Benjamin Flowers, to approximately five developers. The SE team is responsible for the upkeep of the Asset & Compliance Tracking System (ACTS), used mainly by mid-stream and down-stream oil companies.

TimeTracker, written in an older language called PowerBuilder, is an internal desktop management system used by the employees of Ecocion to keep record of company expenses, work hours, current employees, clients, and projects. Other capabilities exist in the original product but are outside the scope of this project. PowerBuilder has limited capabilities that restricted the expansion of TimeTracker and supplied a dated user interface. As a result, it became necessary for TimeTracker to be upgraded through the use of an up-to-date technology stack. ACTS was originally created in PowerBuilder as well, and the SE team is currently working to transfer this to the same technology stack that is to upgrade TimeTracker.

B. Product Vision

Due to the reasons discussed above, TimeTracker was in desperate need of a visual and functional overhaul. The new software is web-based, instead of desktop-based, and has different functionality as requested by the client. TimeTracker now provides an intuitive project management system for internal use by Ecocion employees. A modern technology stack allowed for a visual upgrade to the application. Unnecessary functions have been eliminated and processes have been made simpler, as requested by the client. Employees are able to log work hours and reimbursement requests, as well as manage employees, clients, and projects. This software has been catered directly to the needs of the client.

II. Requirements

The new web application must be able to include existing functionality of the original app, shown in Figure 1. A user must be able to log personal employee work hours, log company expenses and reimbursements, and manage clients, projects, and employees. These must be organized such that each function has its own page or tab in the application. Each of these pages will have a table to display current and past data. The tables should automatically populate the appropriate data according to the selected parameters. A user must be able to add and delete rows from each table, which will update the corresponding database.

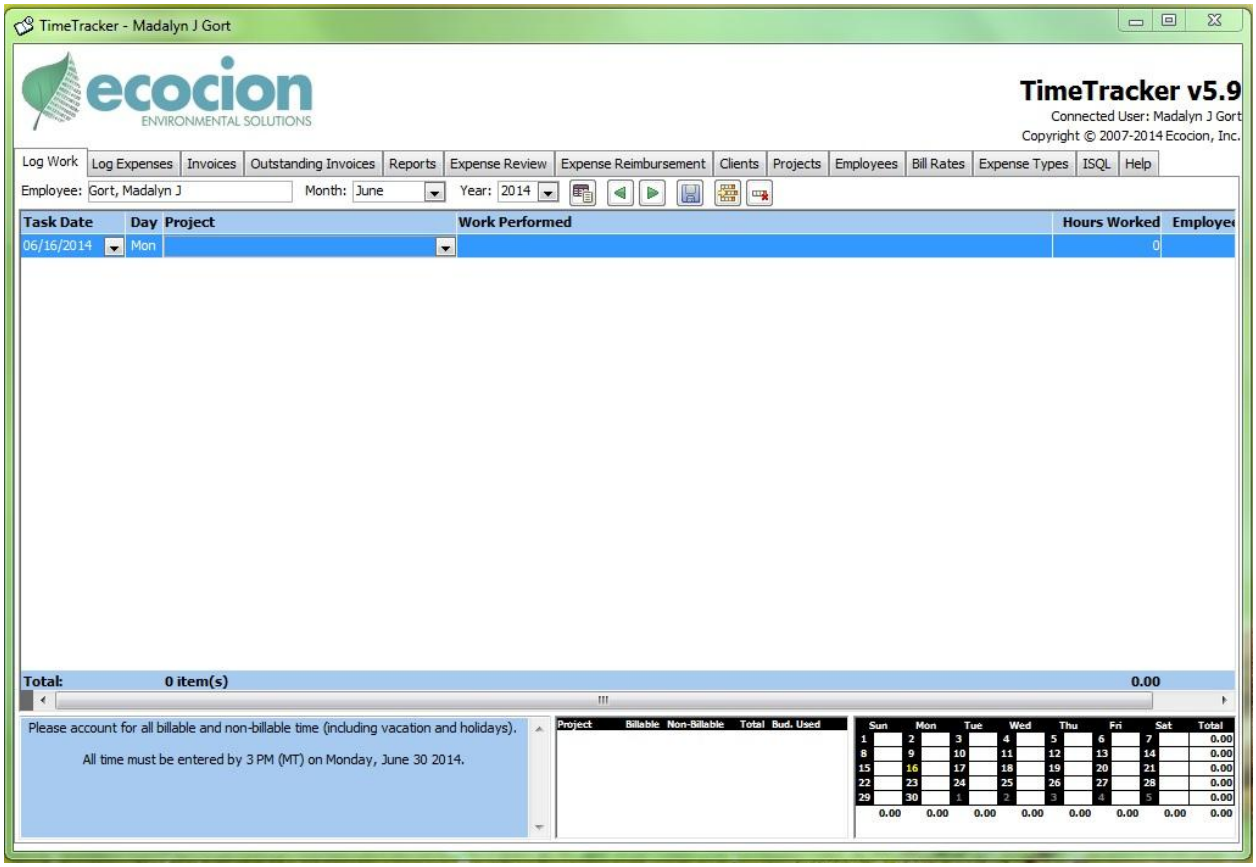

Figure 1: Desktop-based TimeTracker v5.9, written in PowerBuilder

The project's functional requirements include the following:

- The application must be must be web based.
- The application must save each user's data to the database.
- The user must be able to add/delete rows from each table, which must update the database accordingly.
- The application must use databases to store and populate user data.
- The application must implement login security.

- The application must display data based on the current user.
- The application must include the functionality of the desktop version.

The project's non-functional requirements include the following:

- The project must be completed using Ecocion's technology stack and API.
- The project must be coded in Microsoft Visual Studio 2013 using TypeScript, C#, and LINQ as needed.
- The visual standards must conform to the Windows Modern Style.
- The application must be visually upgraded using CSS and HTML5.

III. System Architecture

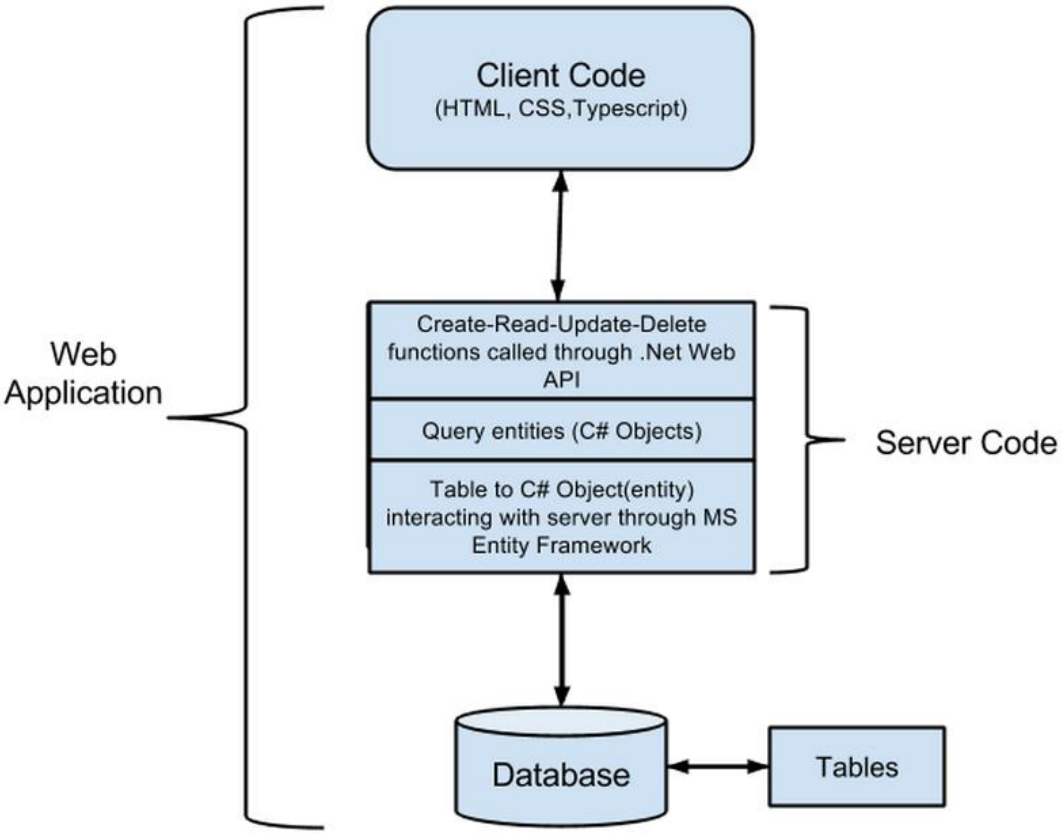

Figure 2: System Architecture Diagram

As shown above in Figure 2, there are 3 key parts of the application: the client code, the server code, and the database. The client code is written in HTML, CSS, and Typescript, a superscript of JavaScript. This creates the user interface portion of the application and is what the user sees when using the product. The server code communicates between the database and the user interface/client code in order to achieve functionality. The Microsoft .Net Web API is used to call the C.R.U.D. functions (Create, Read, Update, Delete), and LINQ is used to communicate with the query entities in the server. Finally, the Table-to-C#-Objects are accessed by interacting with the server through the Microsoft entity framework. Our team is using a top-down approach, which means that we will begin with the client code and work our way down to the server code and database. The database is an Oracle 11g database, which stores all the users' information.

IV. Technical Design

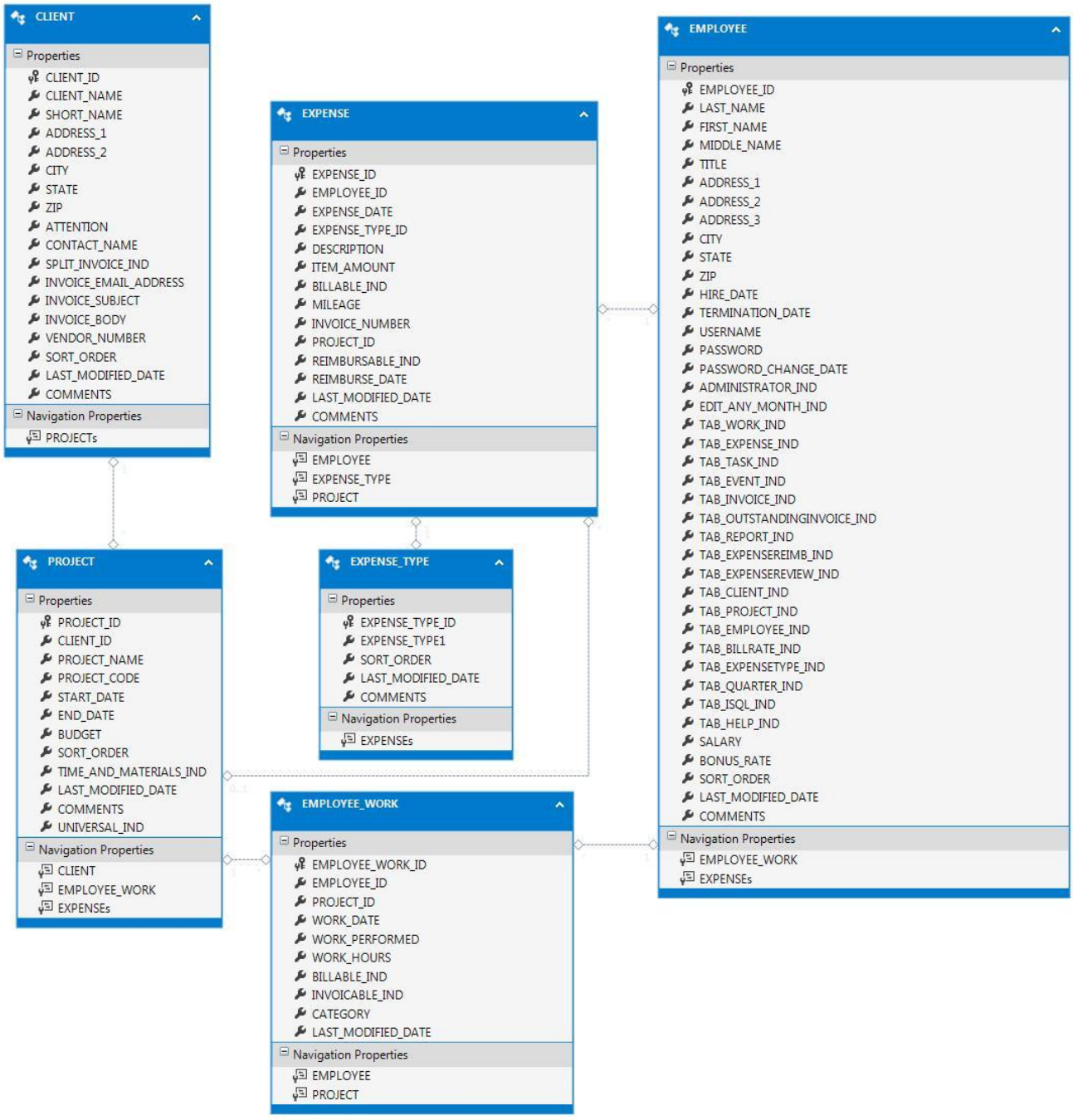

Figure 3: Database Schema Diagram of TimeTracker Database

The tables shown above in Figure 3 contain the layout of the database accessed by the web application. The relationships between the tables are shown by lines attaching to related objects. Often, 2 or more tables were needed to populate a grid on a single tab. Elements from this database were converted to query entity objects in the server code. This database is inaccessible from outside of Ecocion's network and contains personal information related to employees and clients.

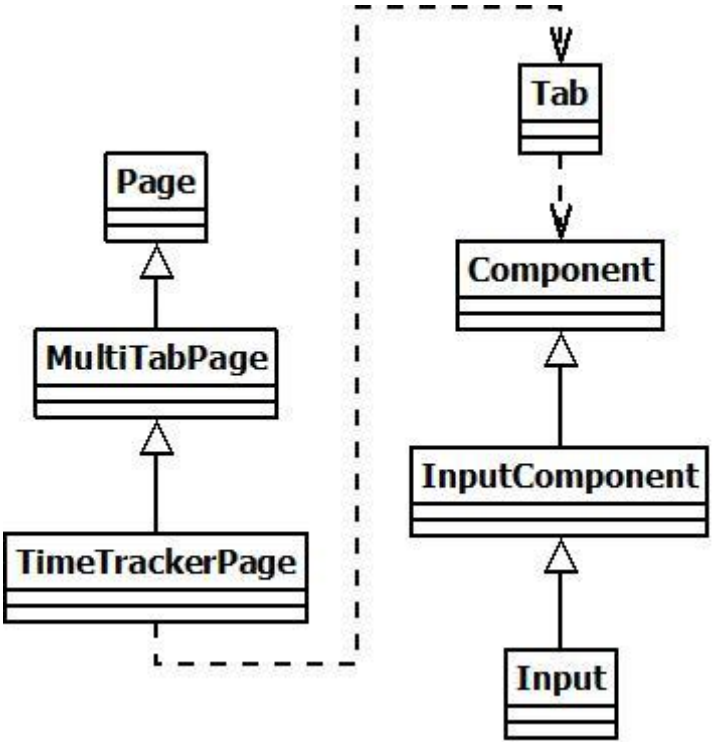

Figure 4: UML Diagram detailing basic layout of API

Figure 4 shows the hierarchy of the elements in Ecocion's API. These elements created the user interface which displays data from the database. The TimeTrackerPage implements a MultiTabPage, allowing the functionality to be expanded. MultiTabPages contain Tabs which contain Components, which ultimately contain data. Components like Grid and Panel, organize the basic structure of the Tab. Inputs, like Textboxes, Dropdowns, and DateTimes are the fields that store and accept new data, according to the current user security privileges.

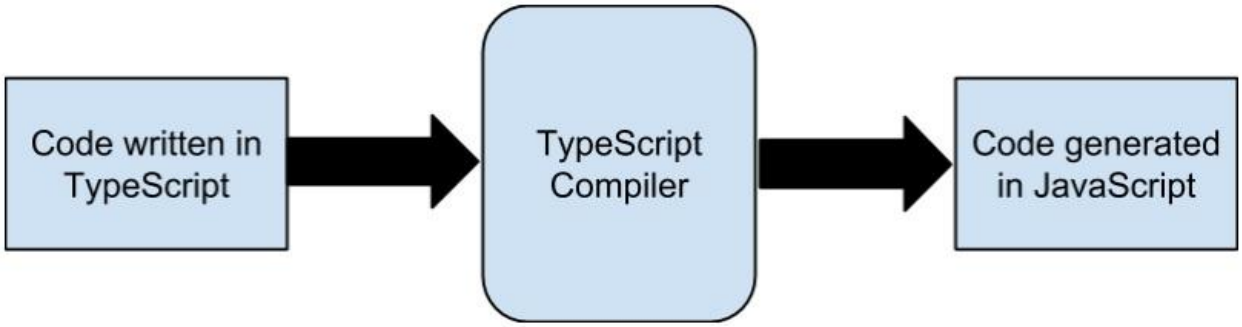

Figure 5: Visual Representation of How TypeScript Works

TypeScript, as shown above in Figure 5, was the one of the chiefly-used languages during the TimeTracker project. TypeScript is a fairly new, open-source programming language developed by Microsoft. It is a superset of JavaScript and contains a compiler that converts written TypeScript into executed JavaScript. Unlike JavaScript, TypeScript provides compile-time errors as opposed to run-time errors which can save time. The Ecocion SE developers prefer Typescript for this reason, and requested that it be used for the upgrade of TimeTracker.

V. Design & Implementation Decisions

We chose to adhere to the technology stack that is utilized by the software engineering department for sake of ease and to ensure seamless integration with the existing software. TypeScript and C#, used by developers in the software engineering team, were determined to be the best solution to use during this project, as this allows us to complete each layer of the web application properly.

When beginning work on the project, it was determined that writing code in the top-down approach would be the most efficient. Referring to Figure 1, this means that work would begin at the top of the diagram with the client code and continue down to the server code.

The Ecocion software team prioritizes working prototypes in addition to using Agile Principles in order to quickly produce working software. Because of this, we chose to employ Rapid Application Development in order to be consistent with this process. We also prioritized working prototypes while adding additional functionality during the course of field session.

In addition to our using Ecocion's API, we chose to alter visual aspects of the new TimeTracker using CSS, as this allowed us to pinpoint aspects of the web application that needed small adjustments. We mainly used CSS to format the "info bar" elements (spacing, alignment, font, etc.), the help tab, and to hide unnecessary buttons.

Despite using Agile methodology during the project, Test Driven Development (TDD) was not used. The software team at Ecocion does not use TDD because the majority of code is in front end, and is difficult to test using TDD. Exploratory, or manual,

testing was used instead by our team and the client in order to ensure the software functions properly. During exploratory testing, testers strive to create a better experience for the user by testing every piece of functionality and attempting to uncover hard-to-find bugs (*A Brief History of Test)*. This allowed several confusing aspects of the software to be corrected quickly.

During the design process of the new web-based application, we determined that the number of things on the “info bar” should be minimized in order to make the application cleaner. On the old application, the arrow buttons were deemed unnecessary, as they moved between months. With a drop down menu, these buttons were redundant. The save and refresh buttons were moved to the top of the page in order to make them stand out to the user and be easily identifiable. Buttons altering inner grid rows, like the add and delete row buttons, were moved to the bottom of the grid to clarify that the intended changes would only affect the currently displayed grid. These changes are shown below in Figure 6.

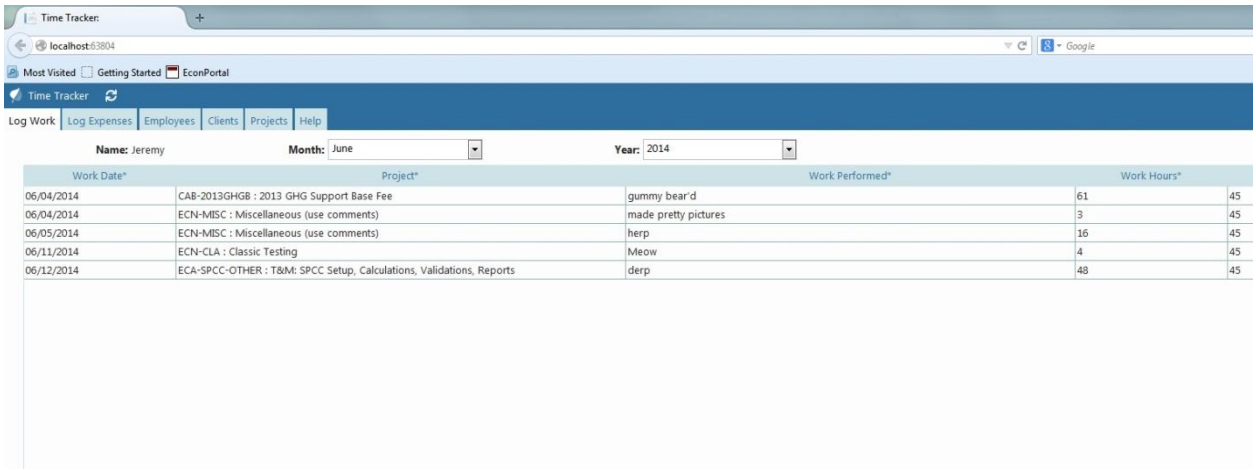

Figure 6: Screen capture of updated TimeTracker

A. Lessons Learned

- Database languages are very helpful. It is useful to be able to communicate directly with the database in order to check that a change made in the API really took effect in the database. Furthermore, being able to use SQL in C# via LINQ is a very useful tool, as it can be easier and more intuitive to implement rather than a raw SQL query.
- Paired programming really works. When a developer is constantly developing software with another developer looking over his or her shoulder, many mistakes can be quickly pointed out, so the developer spends less time stuck on a problem. Two heads are really better than one - a problem that seems impossible to one person can be much more easily figured out with more than one person. In addition, both developers spend less time being distracted and more time doing productive work. While it seemed to be inconvenient during a class where

students had different schedules, paired programming is great for developers that spend the same time at work.

- APIs are extremely powerful. Coding while building upon an existing API can remove redundant code and take advantage of HTML generation. They can increase the amount of time spent working on productive code instead of layouts.
- Don't be afraid to ask for help. If you are a new developer, consider asking a more experienced developer for help before consulting the internet. Senior developers have probably seen the problem that you have encountered and can help you fix the problem faster and more effectively than if you were on your own.

VI. Results

This project is a functional web-based application that communicates with Ecocion's servers to return information to the user. Server code is utilized to communicate with the database using the MS Entity Framework and passes the information to the client code with the .Net Web API. We were able to complete a project that allows the user to alter and insert data as necessary in the web application, like reporting hours, tracking employees, tracking expenses and retrieving client and project information. TimeTracker is now visually upgraded and uses a newer technology stack that matches the current technology stack of the Ecocion Software Engineering developers. TimeTracker will be improved upon in the future as the software developers see fit../

Since this project was to create a prototype of a web-based version of TimeTracker, other developers in the Ecocion office were asked to test the functionality of the app. Several functionalities were improved upon as a result of our thorough testing, such as the ability to "hire" or "fire" employees under the Employees Tab and moving said employee from active to inactive or vice versa.

As a team, we worked closely with the client on-site to determine exactly what the client wanted. Through user acceptance test questions, important questions asked by the client were answered and the software coded accordingly.

Due to time constraints, some functionalities of the product were omitted because the client was more concerned with different functionality. Login security was not included because it was outside of the scope of the project and will be completed internally by Ecocion. We also did not implement a small calendar summary of input hours or other data, since we determined that this would be redundant and would not look aesthetically pleasing to display multiple entries of data per day, limiting the usefulness of this feature.

VII. Appendices

A. User Acceptance Test Questions

Can a user see other user's data?

No. The Log Work tab is only visible to the logged in user. However, the Log Expenses, Employees, Clients, Projects, and Help tabs are visible to all users since this data is public.

Does the tab grid update when parameters are changed?

Yes. When the DropDown menus at the top of certain tabs are changed, the grid reflects the appropriate data.

Can grid data be edited?

Yes and no. Most users have the power to add/delete rows of data from the tabs in addition to editing existing rows. However, if the field is not editable, the user is unable to change that particular field.

Will changing tabs erase unsaved data in another tab?

No. The data will be present until the user attempts to close the application, when they will be reminded of unsaved data. When the save button is clicked, it saves all changes in all tabs to minimize data loss.

Can a user be transferred in between active/inactive employees on the Employees tab?

Yes. If an active Employee is given a termination date, the employee will be transferred to the Inactive Employees section. On the other hand, if an Inactive Employee's termination date is removed, the employee will be "re-hired" and moved to the Active Employees grid.

Does the application save and delete data to the database or only in the application?

The application saves and deletes objects from the database using a series of SQL/LINQ calls. The changes made in the application are reflected accurately in the database.

B. Further Language Information

Further reading about languages discussed in this project can be found at the links below:

PowerBuilder

http://www.sybase.com/products/modelingdevelopment/powerbuilder

TypeScripts

http://www.typescriptlang.org/

C. Sources

"Main." *Ecocion*. Web. 12 June 2014.

"Steve Rowe's Blog." *A Brief History of Test*. Web. 12 June 2014. <http://blogs.msdn.com/b/steverowe/archive/2014/06/04/a-brief-history-of-test.aspx>.