# CHAPTER ONE

# INTRODUCTION

## 1.1: Introduction

Web based applications have evolved significantly over recent years and with improvements in security and technology there are plenty of good opportunities to develop a system as web based application. The Payroll System project is a fully Ajax enabled web application which is developed to solve the manual payroll system. The framework that was used in the project is Ruby on Rails, which is open source full-stack web application framework for the Ruby programming language. Ruby on Rails uses the Model-View-Controller architecture pattern to organize application programming. In this dissertation, we will go through all the steps that were involved in the development process of the system.

## 1.2: Problem Statement

Manual payroll system calculates the employee's salary entirely on paper. If the system is not systematic, searching for particular employee detail is very difficult. In addition, the pay slips for the employee will have to be manually generated. This problem will increase the amount of time to process the payroll transaction. Besides, it is very easy to make mistakes when processing payroll transaction manually, especially mistakes in taxing, which can be very costly. Furthermore, if the detail of the employee is kept on document, the document maybe seen by other people. All of this problem can be a burden for the payroll administrator due to the amount of manual works to be done by the payroll administrator before the pay slips can be given to employee. In order to solve the manual payroll system, a computerized payroll system is essential. Some suggestion has been made to solve the problem that faced by the payroll administrator. For example, a user has to enter their username and password in order to access the system. This will make the system more secure. The payroll administrator also can control the user authority to make sure the data is secure and only can be accessed by the authorized user. Moreover the chances of mistakes are lower because all the calculation is done by the system itself. The data is kept in a systematic way where it allows data to be searched. For example, payroll administrator can search by employee id, employee name, and others. The pay slips for employee will also automatically generated by the system with a mouse click. Payroll administrator can print out the pay slips anytime they want, and the pay slips will display all the details for the calculation of employee's salary. Data modifications are also easy because it can be done with just a mouse click.

## 1.3: Significance of the project

The significance of the project is listed below:

* The Payroll System will help to reduce work load and reduce the risk of information loss.
* The Payroll System will help the payroll administrator to manage the employee’s salary in an efficient manner and increase work performance.
* The Payroll System will able to calculate the salary based on overtime, hourly rate, deduction and allowances according to the data inserted by the administrator.
* User can retrieve and view the records of employee's detail by using the search function from the system or generate various charts.
* The system will be more user friendly with web based application. The system will eliminate the time-consuming and potentially inaccurate method of handwritten notes and manually counted salary.
* The work performance of the payroll administrator will be improved, since using computerized system can save a lot of time.
* The information can be retrieved in a shorter time compared to the manual system. It will also reduce the data input time.
* Payroll administrator can maintain the employee's detail in an easier way by just using the update and delete function from the system.
* The system uses database to store the employee’s detail, which encourages the integration of data and makes data more widely available.

### 1.3.1: Benefits of web based application

There are certain numbers of benefits of the project to be developed as a web based application. The benefits include:

* **Cross platform compatibility**

Most web based applications are far more compatible across platforms than traditional installed software. Typically the minimum requirement would be a web browser of which there are many. (Internet Explorer, Firefox, Google Chrome, to name but a few). These web browsers are available for a multitude of operating systems which includes Windows, Linux, or Mac OS.

* **More manageable**

Web based systems need only be installed on the server placing minimal requirements on the end user workstation. This makes maintaining and updating the system much simpler as usually it can all be done on the server. Any client updates can be deployed via the web server with relative ease.

* **Highly deployable**

Due to the manageability and cross platform support deploying web applications to the end user is far easier. They are also ideal where bandwidth is limited and the system and data is remote to the user. At their most deployable the developer simply need to provide the user a website address to log in to and provide them with internet access.

This has huge implications allowing the developer to widen access to the systems, streamline processes and improve relationships by providing more users with access to the systems.

* **Secure live data**

Typically in larger more complex systems data is stored and moved around separate systems and data sources. In web based systems these systems and processes can often be consolidated reducing the need to move data around.

Web based applications also provide an added layer of security by removing the need for the user to have access to the data and back end servers.

* **Reduced costs**

Web based applications can dramatically lower costs due to reduced support and maintenance, lower requirements on the end user system and simplified architecture.

## 1.4: Payroll System

Todo

## 1.5: Ajax web application

Ajax (an acronym for Asynchronous JavaScript and XML) is a group of interrelated web development techniques used on the client-side to create asynchronous web applications.

In classic web application model, most user actions in the interface trigger an HTTP request back to a server. The server does some processing and then returns a HTML page to the client. This approach makes a lot of technical sense, but it doesn’t make for a great user experience. With Ajax, web applications can send data to, and retrieve data from, a server asynchronously without interfering with the display and behavior of the existing page.

Through Ajax, a web page feels like a desktop application.



Figure 1: The synchronous interaction pattern of a traditional web application (top) compared with the asynchronous pattern of an Ajax application (bottom)

### 1.5.1: Why Ajax is widely used in Web Applications

* They can use a standard web browser, such as Firefox, Internet Explorer or Safari, as their only user interface.
* They don't force the user to wait for the web server every time the user clicks a button. This is what "asynchronous" means. For instance, gmail fetches new email messages in the background ("asynchronously") without forcing the user to wait. This makes an AJAX application respond much more like a "real" application on the user's computer, such as Microsoft Outlook.
* The Ajax engine works within the Web browser (through JavaScript and the DOM) to render the Web application and handle any requests that the customer might have of the Web server. The beauty of it is that because the Ajax engine is handling the requests, it can hold most information in the engine itself, while allowing the interaction with the application and the customer to happen as asynchronously and independently of any interaction with the server.
* They use standard JavaScript features (including the unofficial XMLHTTPRequest standard, pioneered by Microsoft and adopted by Firefox and other browsers) to fetch data in the background and display different email messages or other data "on the fly" when the user clicks on appropriate parts of the interface.
* Usually they manipulate data in XML format. This allows AJAX applications to interact easily with server-side code written in a variety of languages, such as PHP, Perl/CGI, Python and ASP.NET. Using XML isn't absolutely necessary, and in fact many "AJAX" applications don't -- they use the XMLHTTPRequest object to send and receive data "on the fly," but they don't actually bother packaging that data as XML.
* with Ajax, the JavaScript that is loaded when the page loads handles most of the basic tasks such as data validation and manipulation, as well as display rendering the Ajax engine handles without a trip to the server. At the same time that it is making display changes for the customer, it is sending data back and forth to the server. But the data transfer is not dependent upon actions of the customer.

### 1.5.2: Technologies used in Ajax

JavaScript

* Loosely typed scripting language
* JavaScript function is called when an event in a page occurs
* Glue for the whole Ajax operation

DOM

* API for accessing manipulating structured documents
* Represents the structure of XML and HTML documents

CSS

* Allows for a clear separation of the presentation style from the content and may be changed programmatically by JavaScript

XMLHttpRequest

* JavaScript object that performs asynchronous interaction with the server

## 1.6: Methodology

The methodology that was used in the Payroll System is Software Prototyping technique. The Software Prototyping technique refers to the activity of creating prototypes of software applications. A prototype typically simulates only a few aspects of, and may be completely different from the final product. The process of prototyping involves the following steps, which are identifying basic requirements, develop initial prototype, review, and revise and enhance the prototype.

## 1.7: Objectives

The objective of the project is to develop a web based payroll system using Ruby on Rails and Ajax technologies to:

* allow payroll administrator to create new or maintain existing employee details and the salary structure
* organize the entire employee’s salary structure in one database that can only be accessed by authorized administrators
* allow payroll administrator to define the deductions, tax, generate payslip, and generate various charts
* allow employee to maintain his/her own details
* allow employee to view payslip
* allow employee to generate various charts

## 1.8: Project write-up Layout

The dissertation is divided into seven chapters, briefly described as follow:

* Chapter Two: will demonstrate the literature review on traditional Payroll System that was not developed as Ajax web application.
* Chapter Three: Analysis and Requirements Specification reveals the purpose, goal, scope of the Web Based Payroll System and preliminary investigation. It also clearly specifies the functional and non-functional requirements, which identifies the complete specification of requirements for the system development.
* Chapter Four: System Design presents that how the system process flow and the design of the system process for the development of the system.
* Chapter Five: Implementation presents that how the system is being written and developed according to the design that has been developed to lead the implementation.
* Chapter Six: System testing presents the testing strategy and test case to test the Web Based Payroll System and test results of the system.