**Rochester Institute of Technology**

**B. Thomas Golisano College**

**of**

**Computing and Information Sciences**

**Master of Science in Information Sciences and Technology**

**~ Thesis Proposal Approval Form ~**

Student Name : Snedden Gonsalves

Thesis Title : Data Visualization web application

Thesis Area(s): Application Dev. Database √ Website Dev.

(√ Primary area) Game Design HCI eLearning

Networking Project Mngt. √ Software Dev.

Multimedia System Admin. Informatics

Geospatial Other

~ MS Thesis Committee ~

Name Signature Date

Prof Daniel Boggard

Chair

Prof Edward Holden

Committee Member

Data Visualization Web Application  
by  
  
  
**Snedden Gonsalves**  
  
  
  
Project Submitted in partial fulfillment requirements for the degree of  
Master of Science  
in  
Information Technology   
  
  
Committee Chair  
Prof. Daniel Boggard   
Department of Information science and technology  
  
  
  
B. Thomas Golisano College of Computing and Information Sciences  
Rochester Institute of Technology   
Rochester, New York  
July, 2016

Contents

[Abstract 4](#_Toc458107157)

[Introduction 5](#_Toc458107158)

[Problem Statement 6](#_Toc458107159)

[Proposed solution 6](#_Toc458107160)

[Methodology 7](#_Toc458107161)

[Software architecture 7](#_Toc458107162)

[User Flow 10](#_Toc458107163)

[Data visualization/charting library 14](#_Toc458107164)

[Building data visualization functionalities 17](#_Toc458107165)

[Database 18](#_Toc458107166)

[Database Schema Diagram 19](#_Toc458107167)

[Exporting project files 20](#_Toc458107168)

[Technologies Used 21](#_Toc458107169)

[Deliverables 22](#_Toc458107170)

[Time Line 22](#_Toc458107171)

[References 23](#_Toc458107172)

# Abstract

Visualization is communication of information using graphical representation, pictures have been used as a mechanism for communication even before formalization of written language (Ward, Grinstein, & Keim, 2010). With the advent of the digital age more and more data is now formatted in a computer as ones and zeros and the volume of these ones and zeros i.e. digital data has grown exponentially in recent years (Porter).Digital data represented as a graphic is more easily interpreted by the human mind as image interpretation is performed in parallel with the human perceptual system while text analysis is limited by sequential process of reading (Ward, Grinstein, & Keim, 2010).Thus a graphical representation makes it easier to recognize and extract the information present in huge volumes of data which might have been hidden otherwise.

The proposal is creating a web application that lets users create graphics and charts, which is not only user friendly but powerful at the same time. The application would allow raw data to be mapped into interactive visuals. The purpose of the application would be to serve as a data visualization tool, to create custom reports and embeddable code for use in webpages. The goal of the web application is to make a user-friendly GUI for the web platform that finds a balance between being powerful and easy to work with at the same time.

# Introduction

Data visualization in its core has been close to mankind since its simpler times. Right from the early time of map-making and cartography to a more modern use in statistical graphs and data analysis. (Friendly 1,2,3) .More and more data is now represented in digital format and the capacity of digital data has doubled every nine months (Porter).As our ability to store and value data in digital format grows it becomes even more difficult to extract or mine information from the growing volume of data. Data visualization is an important factor of the data mining process as it allows the user to directly interact with the data first hand as opposed to automated techniques (DASTANI 601-622).

The objective of this application is to find a balance between user friendliness and being flexible in its functionalities. The proposal is to create an application which does not simply make ready-made layouts and charts but gives the opportunity to the user to create more basic building blocks of a graph or chart thus making it more powerful than a mere tool that creates predetermine templates for charts.

# Problem Statement

To create a highly customizable data visual or graph on the web a user needs to be adept at web technologies such as JavaScript, CSS, HTML, etc. Even with being well-versed with these technologies creating data visualization could be a tedious task. Although there are many data charting and visualization libraries which are very powerful and help in the data visualization process they involve a learning curve and may not offer the creative flexibility exacted by the user. They also require the user to have some background in programming to be used effectively.

# Proposed solution

Proposed solution is a data visualization tool having a graphical interface which would allow the user to create graphs from scratch using basic building blocks of the chart building process. Thus making it possible to customize the graphs to a higher degree without the need of having a programming background.

The proposed web application would dissect a typical process of creating a chart in a typical data visualization library into smaller processes and reflect these smaller dissected processes on a visual interface. Any chart or data visual is built-upon fundamental building blocks, for instance a typical bar chart maybe dissected into the x/y axes, the data mapped into these axes, the rectangular bars, the labels denoting the bars etc. The proposed web application would recognize the processes to build these fundamental building blocks of a chart. Thus the user could then use these dissected processes to make a chart or a data visual from its base elements. Empowering him or her to make data visuals with a higher degree of customizability.

# Methodology

## Software architecture

### General Design

The proposed application would follow the Model View Controller or MVC design pattern. MVC divides an interactive application into the model i.e. data, view i.e. display and the controller (M. Selfa, Carrillo, & del Rocío Boone, 2016). The model represents the state of the application, the view dictates how this data or state is displayed to the user and the controller can be used to change the model. MVC has demonstrated its benefits for interactive applications allowing multiple representations of the same information, promoting the code reutilization, and helping developers to concentrate on a single application aspect(M. Selfa, Carrillo, & del Rocío Boone, 2016).



MVC Architecture (The model, view, and controller (MVC) pattern relative to the user, 2016)

The **model** is core of the application, it is the data the application is built around. For instance in a simple blogging web application the model might be the posts and the comments. Model in a web application is typically stored separately in a database. In addition to maintaining the state of the application the model can also be used to implement the business rules, for example: in a banking application a single user account cannot be lower than a certain amount. Thus the model is more than just a data store but a gate keeper to perform final validation before the data is updated.

The **view** is how the data is formatted and displayed to the user. In a web application it is typically the HTML and the CSS that is rendered in a browser. The model feeds some data to the view to which then the view formats and renders it before displaying the data to the user. The view essentially provides a user interface to the end user .For instance in the same example of a simple blogging website a model holds the comments and post of user but it is the view that formats and renders this data to be displayed to the user.

The **controller** acts as a link between the view and the model. It handles the input-processing, updates the model, decides which view is to be rendered and when, or when to redirect the page. Again, in our last example of a simple blogging website; a user may update his or hers post, essentially updating the model, in this case the controller deals with handling the user input, processes it and passes it on to the model. On successful update the controller may redirect to a view that shows the post was successfully updated.   
  
The web server for the application would run PHP .PHP is a popular scripting language used for web development ("PHP: What Is PHP? - Manual"). The proposed application will be using a server side framework to implement the MVC design pattern, a framework in the context of web application development defines how a web application would be divided into separate components and how this components would interact with each other. A framework promises higher productivity and shorter time-to-market through design and code reuse (Riehle, 2000). Popular frameworks tend to have huge communities of developers who use and/or maintain the framework and who are quick to support any questions or bugs a user may come across while using the framework ("Web Frameworks: Pros And Cons Of Using Frameworks", 2016). The role of the framework in the proposed application is implementing the MVC design pattern, increasing reusability of the code and to implement some of the common features of a web application such as user authorization, maintaining sessions, database connection libraries, enforcing practices promoting web security etc.

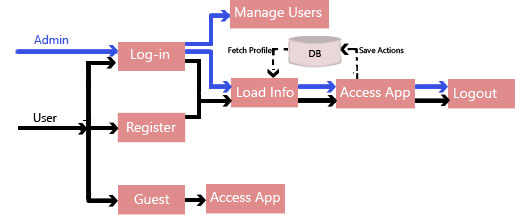
The proposed web application would use a server –side framework called Laravel which is a framework for web applications that run on PHP. Though a relatively new framework Laravel has been growing rapidly and now has become the most popular PHP based framework ("The Best PHP Framework For 2015: Sitepoint Survey Results"). It scores over other PHP based Frameworks such as Zend, CodeIgniter, Symfony on its simplicity while having all the features that a sophisticated framework should have, in fact we may also append components from other PHP frameworks such as Symfony into Laravel. A typical Laravel application implements the MVC components as following:   


Laravel MVC components (laravelbook.com, 2016)

The browser sends a request to the web server say *home/index.php*. The laravel routing engine then calls the controller method which is mapped to this request in this case */index.php* .Now the controller takes over and may interact with the model which is tied to the database. Once the controller fetches or processes the requested information it invokes a view and passes the fetched or processed information to that view. The view then renders the information received in a web browser which is displayed back to the user.

## User Flow

Below is the user flow of the proposed application



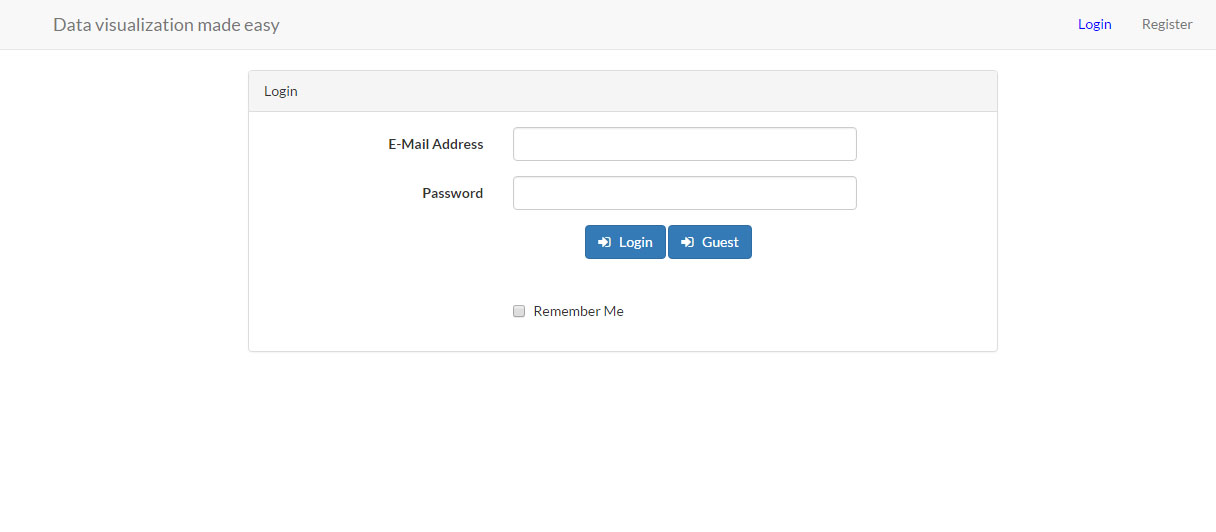
User Flow

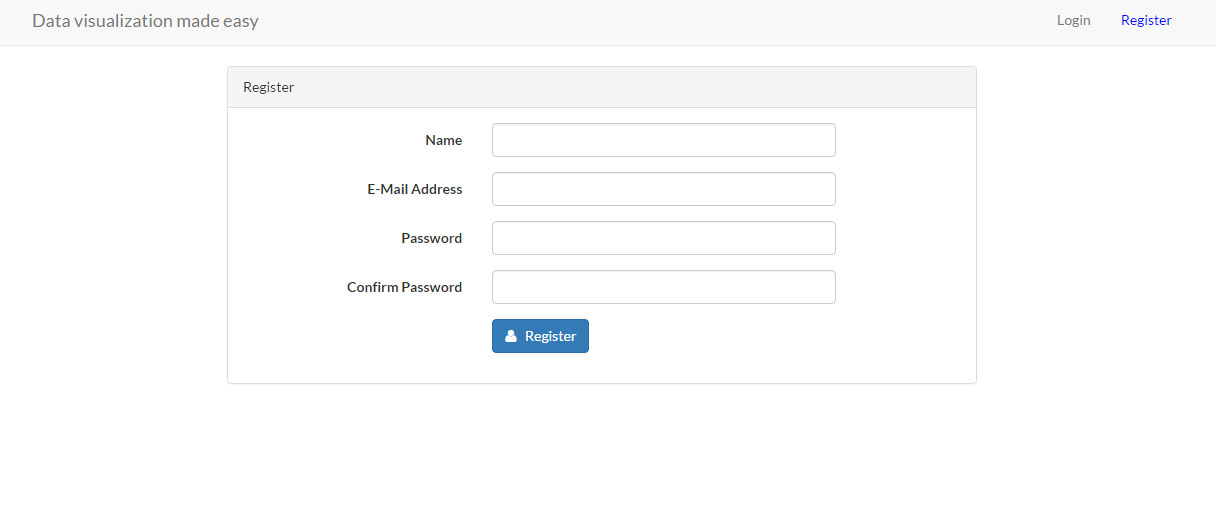
The application will define three types of users namely; administrator, authorized user and guest user. The administrator would be a super user who would have unrestricted access to all the functionalities of the web application including the ability to view, remove/create new users or elevate other users to admin.

A user who has registered and logged-in into the web application is an authorized user. An authorize user could be able to save his profile and preferences .Also an authorize user’s work would be saved into the database thus allowing him to work on the same project in the future.

Below are the rough prototypes of difference views or screens a user would encounter in the proposed web application

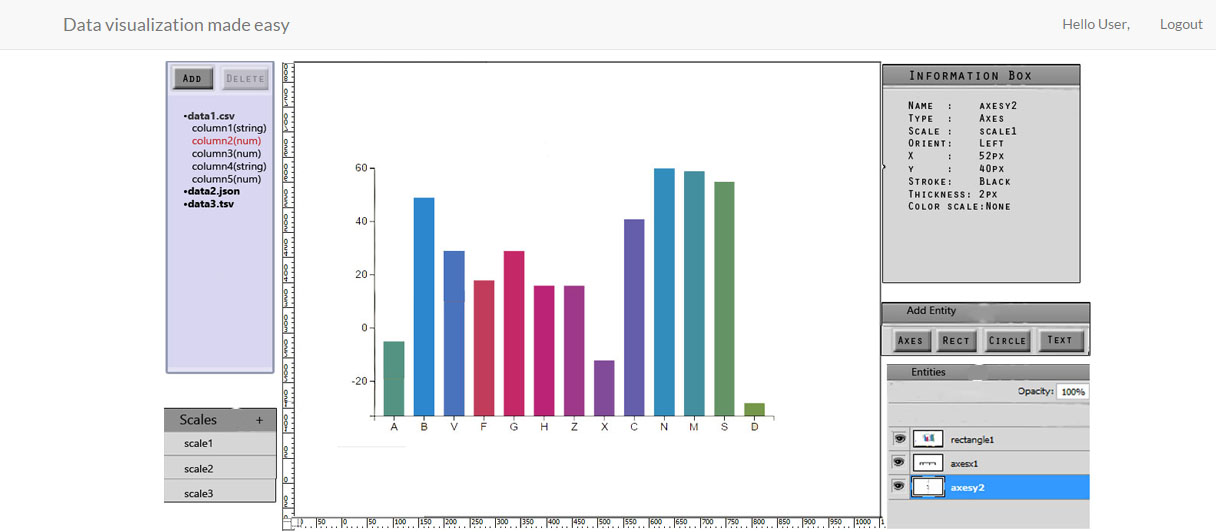
Login Page

  
URI: ‘/’  
  
Access to: Guest user. Authorized and Admin level users are redirected to Application page.  
  
Description: This would be the index page of the web application. A user can login and be thus be elevated to an authorized or admin level user. If a user is already logged in for the current session he or she would be redirected to the application page. If a guest decides to not log in, he could still be able to access the web application as a guest user.

Registration Page

URI: ‘/register’  
  
Access to: Guest user. Authorized and Admin level users are redirected to Application page.  
  
Description: A new user could register as an authorized user using this page. Once successfully register he would be sent to the log in page where he could log in using his or hers’ registered credentials. A successful registration would require a user to input a valid email. User may only create a single account per email ID.

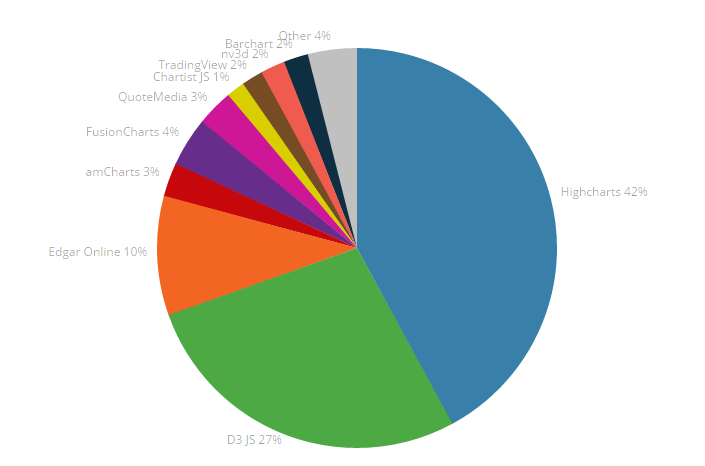
Application page



URI: ‘/application’  
  
Access to: Admin level user has access to all the features while an authorized user doesn’t have access to administrative functionalities namely - managing users. While guest too have access to the same features as an authorized user but a guest user could not be able to save his or her work for future use.  
  
Description: This is the application page. It provides a graphical user interface for creating data visuals. It provides the user a way to upload data using delimiter separated files such as comma separated i.e.CSV or tab separated i.e. TSV .Other supported file format include JSON,XML. A user may export his created project as a bitmap file or copy the embeddable data visual code in his or her clipboard.

Data visualization/charting library

JavaScript is a cross-platform, object-oriented scripting language, it is a small and lightweight language which can run Inside a host environment such as a web browser ("Introduction", 2016). With the advent of web 2.0 technologies the utility of JavaScript has increased and programmers have increasingly created libraries over core JavaScript (Mikkonen & Taivalsaari, 2007). A library is nothing but a suite of functions which could be reused in any related project.

For instance there are a number of data visualization libraries that offer a variety reusable data visualization and charting functions. These libraries help a developer to write code for charts and data visualization without reinventing the wheel. Below is the usage distribution of data visualization /charting libraries and tools at the date of writing this paper according to builtwith.com. Founded in 2007, BuiltWith® is a website profiler, lead generation, competitive analysis and business intelligence tool providing technology adoption, ecommerce data and usage analytics for the internet("About | BuiltWith", 2016). 

Charting library usage distribution (builtwith, 2016)

Let us go through these libraries to briefly explain the features and workings of each starting from the most popular library *Highcharts.* *Highcharts* is an open source library which allows free usage for non-commercial and educational projects. It uses technologies such as HTML, SVG, and VML. The use of VML or Vector Markup Language allows Highcharts to render graphics in older browsers which may not support SVG. *Highcharts* currently supports 20 basic charts which could be combined together to provide higher degree of customizability (Hønsi, 2016)

The second most used data visualization library is *D3, D3* is also an open source library which has minimal restrictions on its usage as per its license .Unlike most other libraries D3 or Data Driven Documents deals directly with the underlying HTML in addition to providing chart templates. D3 is more of a data visualization library than a charting library. This make D3 very powerful but at the same time it adds a steep learning curve for using it. Also it is compatible with IE 9.0+ and all other modern browsers. It may also runs on IE8 with the help of dependent library *Aight*.

Edgar online is an organization with provides financial data and analytics related to market indexes. One of its services is EdgarPro which is-a web based tool which provides charts and quotes for stocks. Moving on to *AmCharts*, *AmCharts* is an open source data charting library which can be freely used for non-commercial purposes. While the free version comes with a mandatory link on created charts pointing to their website the commercial purchase has no such restriction. The library is divided into three categories namely JavaScript charts, JavaScript stock charts, and interactive JavaScript maps each specializing in its self-explanatory type of charts. It has elegant, sophisticated charts and provides an array of different charts subtypes across the three broad categories mentioned above. It has a higher level of abstraction from the underlying technology. *Amcharts* uses both SVG and VML which makes it compatible with older IE browsers, although some of the maps and features might not be support on older browsers.

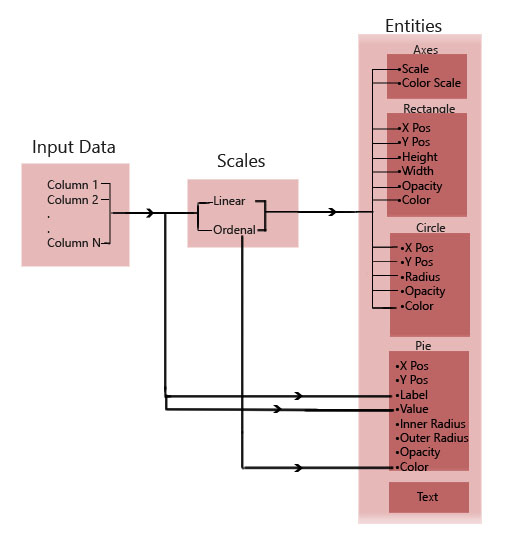
Next in the line is *FusionCharts,* *FusionCharts* is a commercial, open source charting library. The library comes with around a dozen basic charts. *FusionCharts* also provides around thousands of compatible map files which could be downloaded from their website. It has excellent support across browsers, old and new. *FusionCharts* also provide plugins which makes them compatible with other front-end technologies such as jQuery and Angularjs. Some other features include having a wrapper in PHP or ASP.net to support server-side development for *FusionCharts*.

Chartist is a lightweight, open source charting library. One of its prominent feature is its responsive design of the charts which makes the design flexible across screens of different sizes. It has a humble amount of basic charts, but offers several kinds of customizable sub graphs which are compatible with all modern browsers.

NvD3 is an open source charting library built upon D3. It attempts to build reusable charts built with D3. It currently offers 12 customizable basic charts. It support all modern browser starting from IE 9.0.

*Bar chart* and trading view are similar to *Edgar online* who provide quotes and charting to market indexes.

Most of the above libraries provide a high level of abstraction to the underlying technology. Doing so makes it easy to build reusable charts but this decreases the control the user has over building these graphs. The proposed web application emphasis on data visuals and highly customizable charts. The proposed application aims to gives the user creative freedom in creating data visuals and graphs. Libraries which prominently offer pre-made charts limit creativity and customizability of the graphs. *D3* deals directly with the document object model or DOM i.e. the building blocks of a HTML file. This grants the developer a greater control on how he can build different data visual using *D3*. With D3, designers selectively bind input data to arbitrary document elements, applying dynamic transforms to both generate and modify content (Bostock, Ogievetsky, & Heer, 2011). D3 is an apt choice of data visualization library for the proposed web application

Building data visualization functionalities  
Once user flow mechanism is implemented the next step would consist of implementing the functionality of the web application.   
Any data visualization in D3.js starts with having the input data, the data can be a file of delimiter-separated values such as csv or tsv file, a spreadsheet, JSON or XML .The data is then read into D3.js which could be used as an input to a chart.   
  
The process of building a chart in the web application is aggregated as shown below:   
  
The columns of the input data would be associated to a scale. Scales in D3.js are functions which map input data or domain to output abstract or range which can be used as a reference to draw visuals such as graphs and charts (Swizec).The web application would use two kinds of scales: Ordinal scales to map categorical data e.g. labels and Linear scales to map linear data e.g. height.

These scales once associated with an input column of the data can then be in turn be associated with an Entity. Entities as defined within the proposed application are the visual graphics of the web application which could be used as building blocks of a chart.

The web application would have 4 kinds of Entities namely: Axes, Rectangle, Circle, Pie and Text. A scale could be associated with certain attributes of an entity say width for a Rectangle entity or radius for a Circle entity which would then reflect the values of the data column associated with the scale.

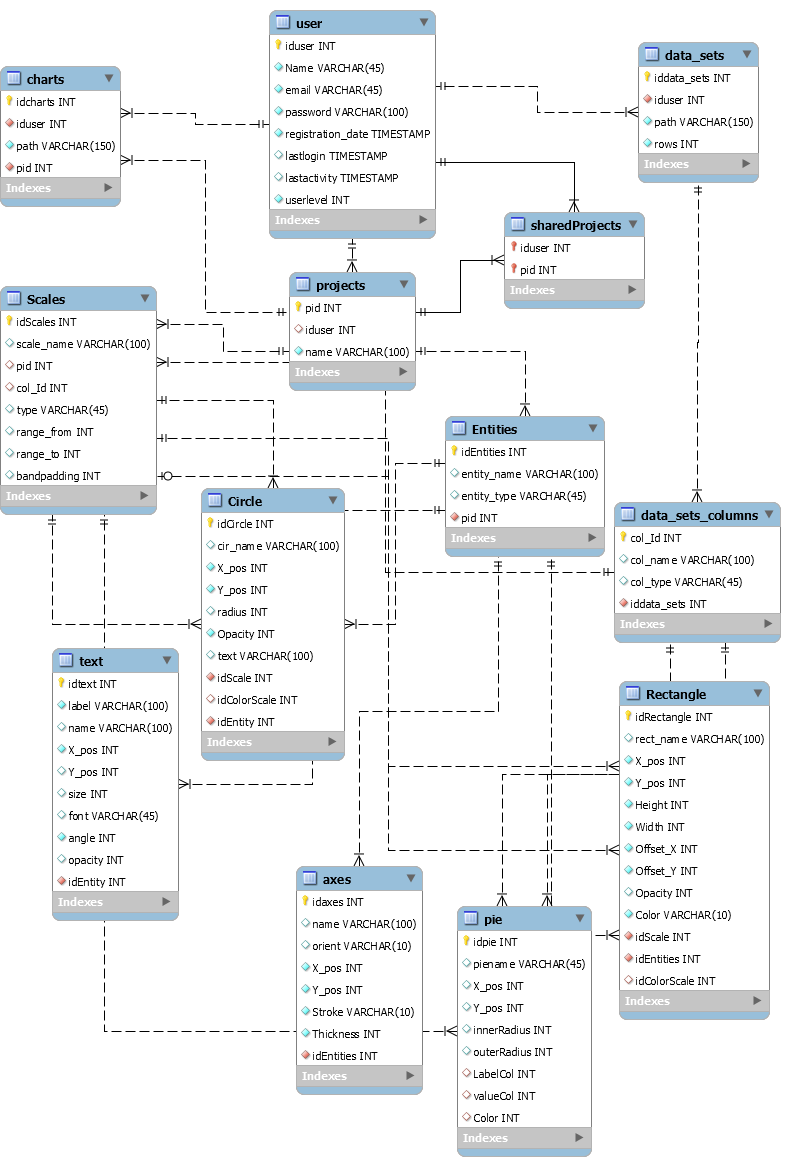
## Database

The first step in choosing a database is choosing whether the proposed application would go for a SQL based Relational database or a NoSQL database.

For SQL based relational databases some of their advantages are that they are better with primarily structured data, complex/nested transactions, have powerful query language for complex operations, they scale up. (Issac, 2016)

On the other hand for NoSQL databases are better with primarily unstructured data or loosely structured data, they deal better with high velocity data, could better maintain high volume of data, they scale out. (Issac, 2016)

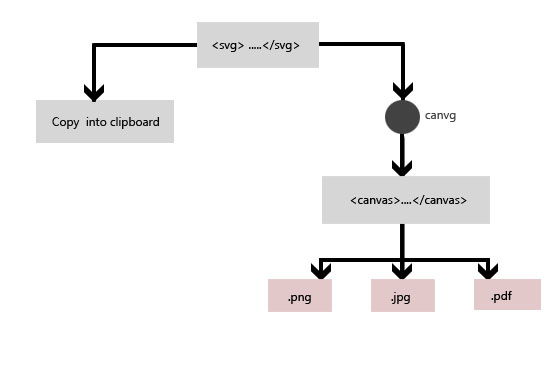
As the proposed web application would primarily have highly structured data, generate moderate amount of data, require complex queries to be executed on the database layer and scalability would not be an issue due to relatively lower volume of data a relational database would be an apt choice for the web application.  
  
The leading contenders for relational databases are Oracle, SQL server and MySQL ("DB-Engines Ranking - popularity ranking of database management systems", 2016). Although there are number of differences between these RDMS there are two unique aspects to MySQL firstly it was not originally developed for commercial use and secondly it is an open source and free to use although support is commercial.   
  
The proposed web application would use MySQL for the database system. MySQL would be a good fit for the proposed web application as it is straight forward and it has proven to out-perform other RDMS in regards to web applications. (Suehring 18)

Database Schema Diagram

## Exporting project files

The user can export his or her work i.e. the data visual he or she creates in two ways .The user could either copy/paste the embeddable svg along with a reference for D3.js as a dependable library allowing to embed the created data visual in some other html page or the user could export the data visual as a file which the user could then use offline.

D3.js uses scalar vector graphic or SVG to render graphics on a web page. SVG is a XML based technology that renders vector graphics on a browser. As of the day of writing this proposal D3.js doesn’t have an intrinsic functionality to convert the SVG into bitmap or pdf files. The proposed application though provides a feature where the user can export his or her work into bitmap such as JPEG or PNG format. Thus to export the created data visual we first convert the underlying SVG into canvas which is the scalar equivalent of SVG .The conversion is done by using a JavaScript library canvg. Canvg takes a svg file as input parses it in JavaScript and renders the svg file as a canvas element. The canvas element can then be exported into a bitmap file format.



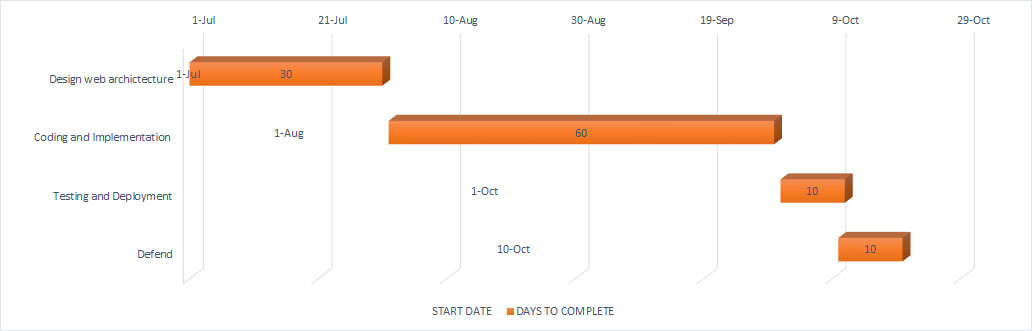
# Technologies Used

1. D3.js  
   D3.js is a JavaScript library created by Mike Bostock for data visualization using JavaScript and svg. It has become the tool of choice for building data visualization on the web platform.D3.js was designed to give programmers a lower level toolset for building data visualizations thus provided a higher control over the technology (King).
2. PHP  
   PHP is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML ("PHP: What Is PHP? - Manual").PHP was created by Rasmus Lerdorf in 1994 as a simple set common gate interface binaries written in C. Since then PHP has evolved as a main stream server side language, more than forty million websites run on PHP today ("PHP Usage Statistics", 2016).
3. MySQL  
   MySQL is an open source SQL database management system, is developed, distributed, and supported by Oracle Corporation ("Mysql :: Mysql 5.7 Reference Manual :: 1.3.1 What Is Mysql?").
4. Laravel  
   Laravel is a free, open-source PHP web framework, created by Taylor Otwell. Laravel implements the Model View Controller or MVC design pattern. It is a more recent framework with its first beta release in 2012. Since then it has steadily rose in popularity and now has become one of the most used PHP based framework.
5. Git  
   Git is an actively maintained open source project for version control originally developed in 2005 by Linus Torvalds. Git is a widely used version control software, it is very powerful and efficient. It has well documented user commands .Also it has several Graphic user interface programs which make using GIT even easier.
6. GitHub  
   Git hub is an online repository hosting service for Git .Users can freely create and use public repositories while private repositories come with a monthly fee. A GitHub repository is an entity where the user stores his/hers project which could be accessed by other authorized users in an efficient way.
7. Bootstrap  
   Bootstrap is the most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web (Mark Otto, 2016).It is a front end framework, frontend in the context of a web application is the user interface i.e. in most cases-the browser. Bootstrap broadly provides three reusable components namely the JavaScript plugins, the CSS and Glyphicons. Glyphicons are browser friendly icon fonts which are generally licensed, however bootstrap provides a number of free to use glyphicons.

# Deliverables

1. Source Code
2. User’s Guide and Technical Documentation.
3. Project Report
4. Working online version

# Time Line



# References

* Swizec, Teller. *Data Visualization With D3.Js*. Birmingham: Packt Publishing, 2013. Print.
* King, Ritchie S. *Visual Storytelling With D3*. Upper Saddle River, N.J.: Addison-Wesley, 2015. Print.
* Friendly, Michael. "A Brief History Of Data Visualization". (2006): 1,2,3. Print.
* DASTANI, MEHDI. "The Role Of Visual Perception In Data Visualization". *Journal of Visual Languages & Computing* 13.6 (2002): 601-622. Web.
* Porter, James. "An Historical Perspective Of Disk Industry". N.p., 2016. Web. 26 May 2016.
* "PHP: What Is PHP? - Manual". *Php.net*. N.p., 2016. Web. 13 June 2016.
* "Mysql :: Mysql 5.7 Reference Manual :: 1.3.1 What Is Mysql?". *Dev.mysql.com*. N.p., 2016. Web. 13 June 2016.
* "The Best PHP Framework For 2015: Sitepoint Survey Results". *SitePoint*. N.p., 2015. Web. 24 June 2016.
* builtwith,. (2016). Retrieved from <http://trends.builtwith.com/javascript/charting>
* *PHP Usage Statistics*. (2016). *Trends.builtwith.com*. Retrieved 23 July 2016, from <http://trends.builtwith.com/framework/PHP>
* Mark Otto, a. (2016). *Bootstrap · The world's most popular mobile-first and responsive front-end framework.*. *Getbootstrap.com*. Retrieved 23 July 2016, from <http://getbootstrap.com/>
* Hønsi, T. (2016). *Chart types | Highcharts*. *Highcharts.com*. Retrieved 23 July 2016, from <http://www.highcharts.com/docs/chart-and-series-types/chart-types>
* laravelbook.com,. (2016). Retrieved from <http://laravelbook.com/images/laravel-architecture/laravel-mvc-components.png>
* *The model, view, and controller (MVC) pattern relative to the user.*. (2016). Retrieved from <https://commons.wikimedia.org/wiki/File:MVC-Process.svg>
* Suehring, Steve. *Mysql Bible*. New York: Wiley, 2002. Print.
* Issac, L. (2016). Retrieved 5 August 2016, from <http://www.thegeekstuff.com/2014/01/sql-vs-nosql-db/>
* *DB-Engines Ranking - popularity ranking of database management systems*. (2016). *Db-engines.com*. Retrieved 5 August 2016, from <http://db-engines.com/en/ranking>
* Ward, M., Grinstein, G., & Keim, D. (2010). *Interactive data visualization*. Natick, Mass.: A K Peters.
* M. Selfa, D., Carrillo, M., & del Rocío Boone, M. (2016). A Database and Web Application Based on MVC Architecture.
* Riehle, D. (2000). Framework Design A Role Modeling Approach. Retrieved from <http://e-collection.library.ethz.ch/eserv/eth:23315/eth-23315-02.pdf?pid=eth:23315&dsID=eth-23315-02.pdf>
* *Web Frameworks: Pros And Cons Of Using Frameworks*. (2016). *1stwebdesigner.com*. Retrieved 7 August 2016, from <http://1stwebdesigner.com/web-frameworks/>
* *Introduction*. (2016). *Mozilla Developer Network*. Retrieved 7 August 2016, from <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Introduction>
* Mikkonen, T. & Taivalsaari, A. (2007). Using JavaScript as a Real Programming Language, 3. Retrieved from <http://www.activemode.com/webroot/Workers/ActiveTraining/Programming%5CJavascript_AsProgrammingLanguage.pdf>
* *About | BuiltWith*. (2016). *Builtwith.com*. Retrieved 19 August 2016, from <http://builtwith.com/about>