ToDone 2 Final Report

User Manual

ToDone is a task manager acting as a personal E-assistant in daily life. In the modern society, the pace of life has kept increasing. As people became busier, it seems more significant for the development of such a personal task manager. Study has shown that a clear plan is beneficial for working efficiently and effectively. As the name has implied, *ToDone* is made to be so simple that it focuses on its key functions: give users a list of tasks, and make all of them DONE.

ToDone 2 is a comprehensive update of the *ToDone* application. Many new features have been introduced and a few bugs have been fixed. Also, Twitter Bootstrap has been applied for better page formatting and display.

(It should be noted that ToDone *and* ToDone 2 *should refer to the same thing, the application that is being talked about, in this user manual.)*

ToDone is built on **Ruby on Rails** as back-end framework and **Bootstrap** as front-end framework. The recommended version for your Rails installation is 5.0.1, while the least requirement is 4.4+. The Bootstrap version is 3.3.7, however, you are NOT expected to install it on your server since CDN service has been used for ease.

Special Notice: This project is under GNU General Public License (GPL) 3.0, which means you can redistribute it and/or modify it or any part of it. However, all such derivative work MUST be under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

There may be security issues or stability issues when using this system. This program is distributed in the hope that it may become useful, but WITHOUT ANY WARRANTY.

Below, a few key features of this system will be highlighted:

- **Create/ Show / Modify / Delete (CRUD) *Categories***

A collection of events are called a category. Each category has two attributes, a name (string) and a description (text).

When a new category is created, its name box cannot be left blank. After a category has been created, events can be added to them. In other words, user has to create a category prior to creating an event.

Special Notice: After a category has been deleted, all of the events in that category will also be deleted simultaneously.

A window for confirmation will pop up when user tries to delete a category to avoid errors of operations.

- **Create/ Show / Modify / Delete (CRUD) *Events***

Each task, matter or business in *ToDone* is called an *event*. Each event has four attributes, namely, *title*, *time*, *place* and *details*. The first two (title and time) are compulsory when creating a new event. However, the remaining two are optional.

In the index page, user can see a table of all the events from all categories. However, user can only see the events belonging to that category in the page for a certain category.

Each line represents an event with the information of title, time and place. These are followed by three links, to *show details*, *edit event* and *delete event*.

A window for confirmation will pop up when user tries to delete an event to avoid errors of operations.

While the details of an event cannot be shown in the table of index pages, please click "*Show Details*" to look for it.

- **Basic HTTP Authentication**

When user try to create / edit / delete any event or category, an HTTP authentication window will pop up and require the user to log in with username and password.

Notice: The **default** username and password is "**user**" and "**123**".

- **Unified Navigation Bar**

The navigation bar system from Bootstrap 3 has been adapted in *ToDone 2*. This application has a unified navigation bar at the top of all pages.

The left-most element of the navigation bar can redirect the user to the main page, while the else points to respective category index page.

Hope you can enjoy using ToDone 2 and please do not hesitate to contact us when you find any problems or would like to offer any advice for improvement.

Niu Yunpeng
neilniuyunpeng@gmail.com

January 2016

My accomplishments

Compared to the last project (my blog system), this project has applied many existing good-quality and well-known framework, such as *Ruby on Rails 5.0.1* and *Bootstrap 3.3.7*. The application of these popular frameworks reduces the boring duplicate part of the development circle, and makes the working much more fun. I have learnt a lot from this project, which can be summarized into a few key points:

- **DRY (Don't' repeat yourself)**

One of Ruby on Rails main philosophy is DRY, which asks the developers never repeat doing anything. This makes the code more readable and avoid unnecessary errors.

On the one hand, use partial files to represent the common part of multiply pages. For instance, the form for creating / editing a category or event may appear in different pages. Therefore, we can put it in a separate file whose name begins with the underscore "_" prefix. My own experience told me this can be quite beneficial. I tried to copy-and-paste the segment of codes here and there at the beginning. This leads to a few mysterious errors and it is really hard and painful to detect them.

On the other hand, avoid exploring yourself if others have gone through yourself. Rails is so well-developed that makes it also so complicated. When any problems have been encountered in the development process, developers should always refer to the official guide for help or search on the Internet to see if the same situation has happened to anyone else. As Ruby and Ruby on Rails on both seeking a best way to develop applications, the development process can be quite similar for different projects. So most likely, other have confronted the same problem and it has been solved.

- **Think from the user's perspective**

A successful application will be widely used by many users, which may be a diverse grouping. This means the user may not think in a similar to what you are thinking about when developing this application. Therefore, it is important to think about what users may do and what are the potential consequences.

A good examples has occurred when I developed *ToDone 2*. In the *Chinese New Year Update* v2.0, all of the categories will be shown in the navigation bar. However, this seems to invalidate the check for creating a new category. Whenever the use leaves the name box of a new category as blank, the application will crash when trying to detect the error.

The reason for this bug has been found at last. When saving a new category is not successful, the user will go back to the "New Category" page. However, there is no name for this new category. The navigation bar will crash since it cannot find such variable for the last element in the array of categories. The error checking for the form does not help since the rendering of navigation is done anyway. The errors are only displayed instead of being prevented to happen.

I had to admit that *Ruby on Rails* is a wonderful platform for developing web applications. Although it may be painful to understand its complicated mechanism at first, it will be much more fun after that. Developers will definitely feel a sense of achievement when they are using *Ruby on Rails* to develop their own web applications.