Exercise 1

1. Click on the "About" link in the navigation bar. What code in the application just got executed? Hint: Look at the address bar in the browser.
   1. It executed code resembling Homecontroller.about. What I mean is that it went to the homecontroller and looked for an action named about. If it matched to an existing action, it would execute that code. If that didn’t match any existing action, it would throw an error.
2. In the address bar of the browser, add /home/about to the end of the URL (after the port number) and hit enter. Explain the process the MVC framework takes to result in the page displayed. How does the MVC framework know what to display based upon the URL it receives?
   1. It first looks at the /home and looks for a matching controller, in this case the homecontroller. If it matches to a controller, it follows by checking if there is anything left in the url. If there is, then it will try to match the next /something to an action; in this case /about. After that, if there is anything left, that value defaults to id. As for the view, that is returned by the action from the controller. If the action is missing, it should default to a default action which returns a default view. That’s how the mvc framework knows what to display. It goes through a check of /controller/action/id and matches relevant info up.

Exercise 2

1. The ActionResult About() within the homecontroller returns a view which in essence is the page.
2. The controller generates the actions that can be used within that page.
3. The view is within the specified controller folder; i.e. Home for the HomeController; within the views folder.
4. The way the views are set up is that all the views are within a views folder. When you go inside that, you have specified controller folders. That means if the call is coming from a controller, the MVC framework can just look for the appropriate controller folder. Within the specified controller folder are all the views that have been created and the framework will try to match to an existing page matching to the action that is calling the view; i.e. The about.cshtml view for the about command. There is also a shared folder for shared views, like the basic layout or the error page, which is not necessarily exclusive to any 1 controller.

Exercise 3

1. Server Error in '/' Application. The view 'Index' or its master was not found or no view engine supports the searched locations.

We see this because there we didn’t make a view to be used by index so it has no view it can use which confuses the program because the index is supposed to return a view.

1. Assuming you meant to create an index view, the error disappears and now we get a page that just says index on it. If not, the same error is there because we only have an index action and there is no view that it can use. Alternately, we could change the action name to Person so that the view will now connect to the proper person action if the view is also named Person. That is the change I made.
2. The page properly displays all the edits made in the viewbag.