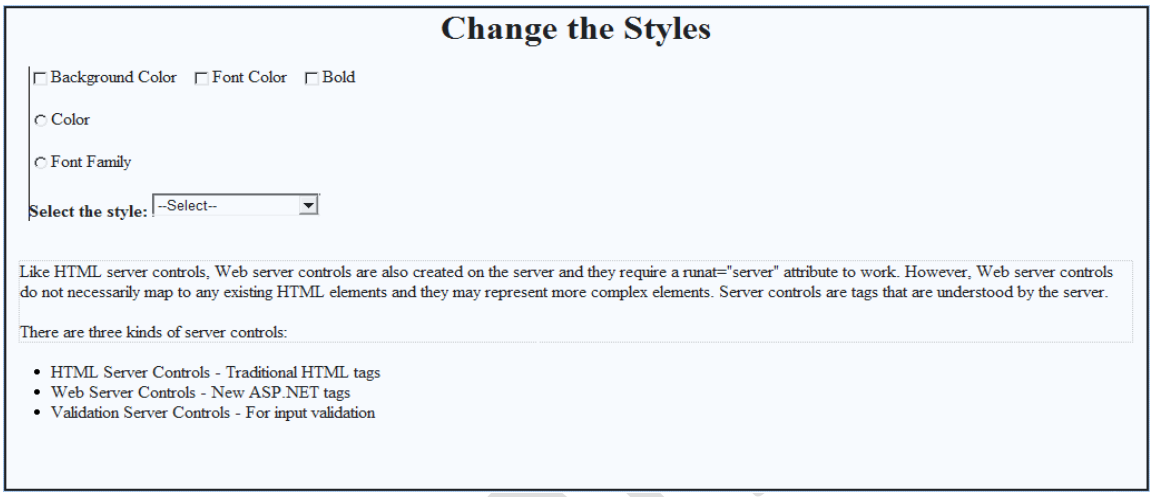
Assignment 1:

Title: Create a web application to change the Background color, font color and font family based on the selected styles. Bind the font styles and colors to single dropdownlist**.**

Description:

1. Create a new Asp.net Web Site.
2. Design the web page as follows. Drag and drop the following controls displayed in the picture and write some static text to which the following styles will be applied.



3. Set the properties for all controls as:

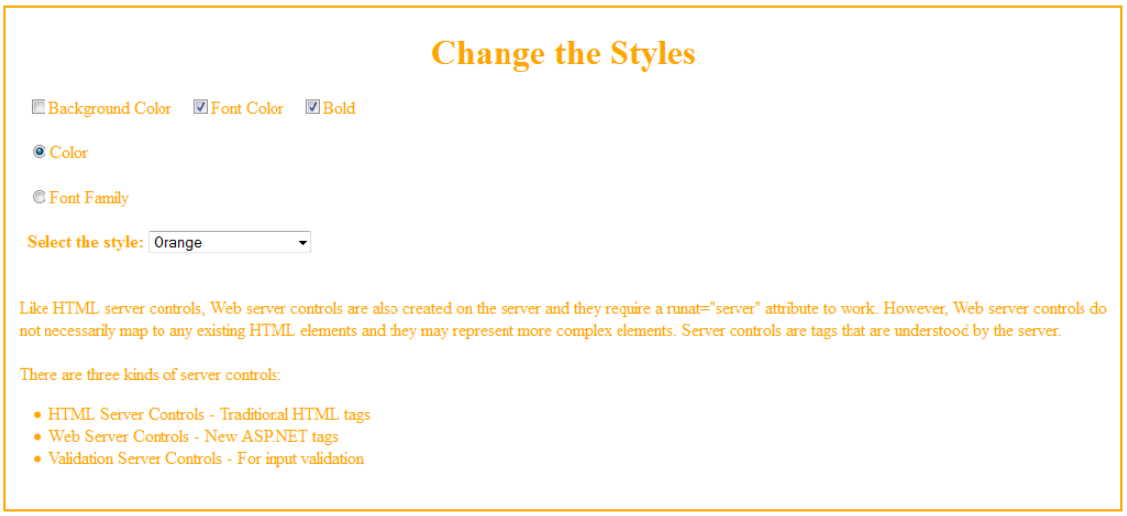
Control	Text	Name	Additional Properties
CheckBox1	Background Color	chkBackColor	AutoPostBack : True
CheckBox2	Font Color	chkFontColor	AutoPostBack: True
RadioButton1	Color	rbnColor	AutoPostBack: True GroupName: Fonts
RadioButton2	Font Family	rbnFontFamily	AutoPostBack: True GroupName: Fonts
DropDownList		ddlFont	AutoPostBack: True AppendDataBoundItems: True Width: 150

4. Create a list of type <string> for color and Fontfamily. Add the values (colors & fonts) to the collection.
5. Handle the CheckedChanged event for RadioButton 'rbnColor' in which check whether the 'rbnColor' is checked or not. If checked then bind DropDownList 'ddlFont' to list of colors. Else if 'rbnFontFamily' is checked then bind 'ddlFont' with list of FontFamily.

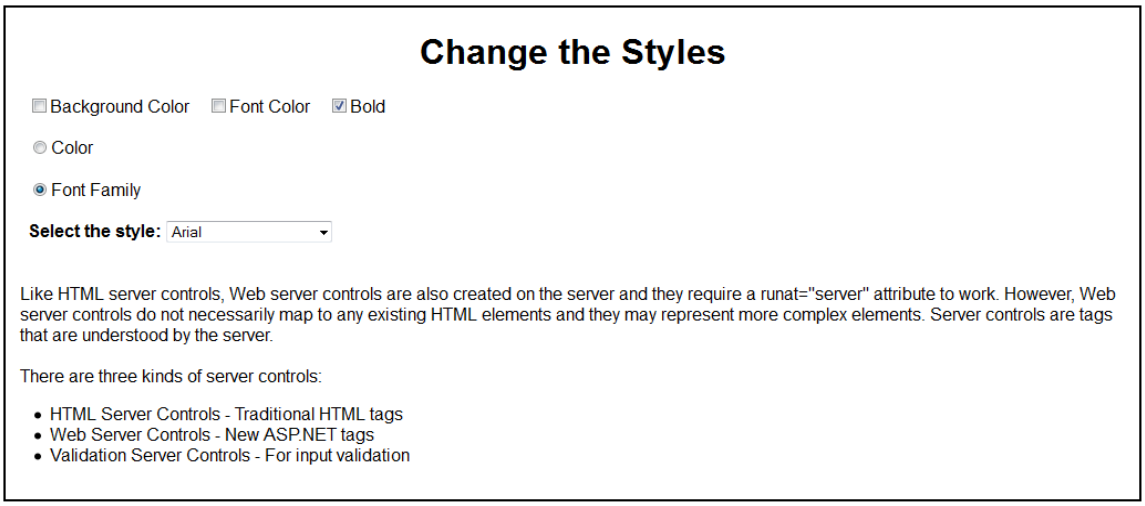
6. Handle the SelectedIndexChanged event for DropDownList 'ddlFont' in which check whether the radiobutton 'rbnFontFamily' and checkbox 'chkFontColor' are checked or not. If 'rbnFontFamily' is selected then change the font family of the body and if 'chkFontColor' is checked then change the font color of the body.
7. Handle the CheckedChanged event for the checkbox 'chkBold'. If 'chkBold' is checked then the text in the body should be added with an attribute 'font-weight: bold'. Else 'font-weight: normal' is added.
8. Handle the CheckedChanged event for both the checkboxes 'chkBackColor' and 'chkFontColor'. For 'chkBackColor' event make chkFontColor.checked = false and for 'chkFontColor' event make chkBackColor.Checked = false.

Sample I/O:

If Font Color, Bold and Color are checked then output would look like:



If Bold and Font Family are checked then output would look like:

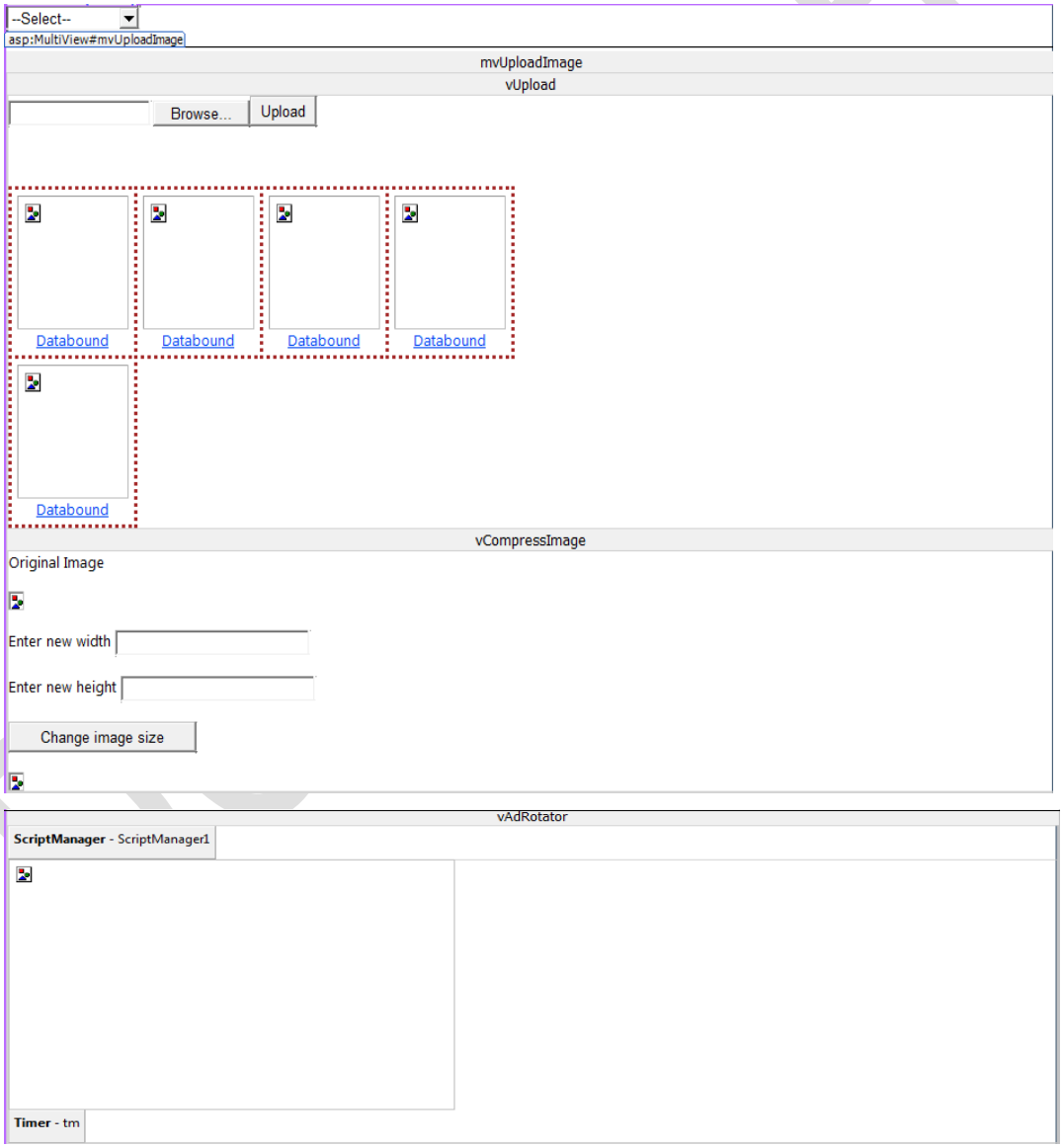


Assignment 2:

Title: Create a web Application using multiview control. The view should change based on the selected option in the dropdownlist. One view should contain the fileupload control to upload the images, save them in the image folder of the project and also display all the uploaded images. The second view should display an image and compress the image based on the width and height which will be given manually and store that image in the Image folder only once. Third view for rotating images using adrotator.

Description:

1. Create a new Web application.
2. Design the web form as shown below:



3. Set the properties for all the controls as follows:

Control	Text	Name	Additional Properties

DropDownList		ddlType	AutoPostBack : True
Multiview		mvUploadImage	
View1 FileUpload Button DataList	 Upload 	vUpload fuImage btnSubmit dlImages	 RepeatColumns=4
View2 Image1 TextBox1 TextBox2 Button Image2	 Change Image Size	vCompressImage imgOld txtWidth txtHeight btnCreateThumbNail imgNew	 Width:167px Width:167px
View3 ScriptManager UpdatePanel AdRotator Timer	 	vAdRotator ScriptManager1 SlideShow AdRotatorFlowers tm	 UpdateMode: Conditional Height:250px Width:400px AdvertisementFile:”~/Fl owers.xml” Interval:2000

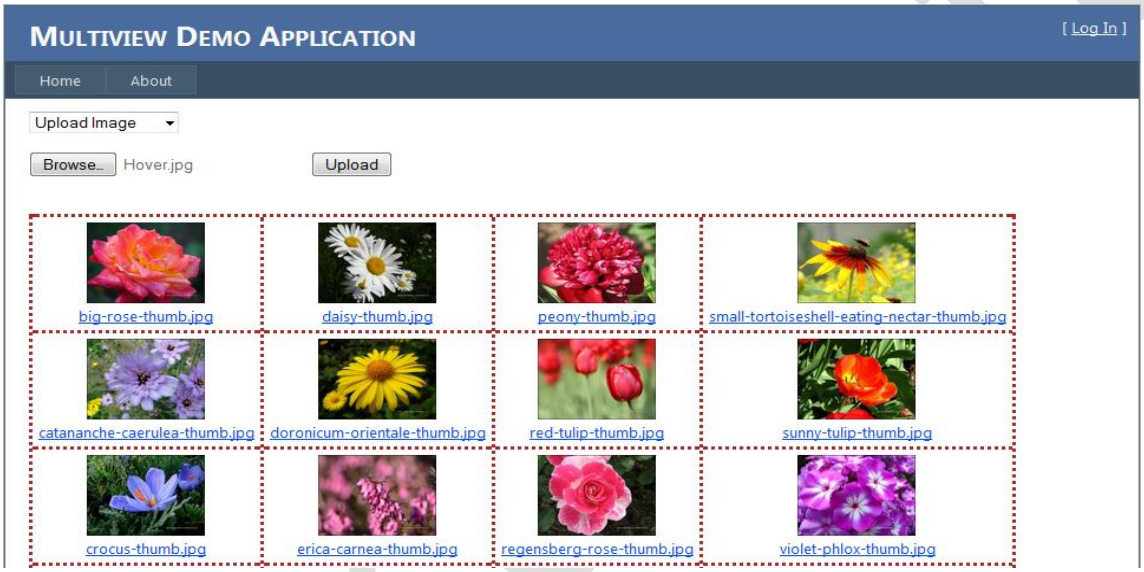
4. Right click on the project → Select New Folder and add few images of flowers
5. Right click on the project → Add New Item →select XML File from the templates and name it as Flowers.xml. Add the Image url (Images you want to rotate) in the XML file. The images below should contain in FlowerImages Folder.
6. In the design view drag and drop 'AdRotator' to rotate the images asynchronously binding the Flowers.xml to the 'AdvertisementFile' property of it and Timer to mention the time interval to change the image in the update panel (Ajax control). Also drag and drop Script Manager to register the controls to Ajax.
7. Define a method 'BindDataList' in that create a DirectoryInfo to get image path and get files from the directory using Fileinfo. Add items from files to the collection of arrayList. Bind the arrayList to DataList.

On the page load event make multiview ' mvUploadImage' visible as false and call the method 'BindDataList'.

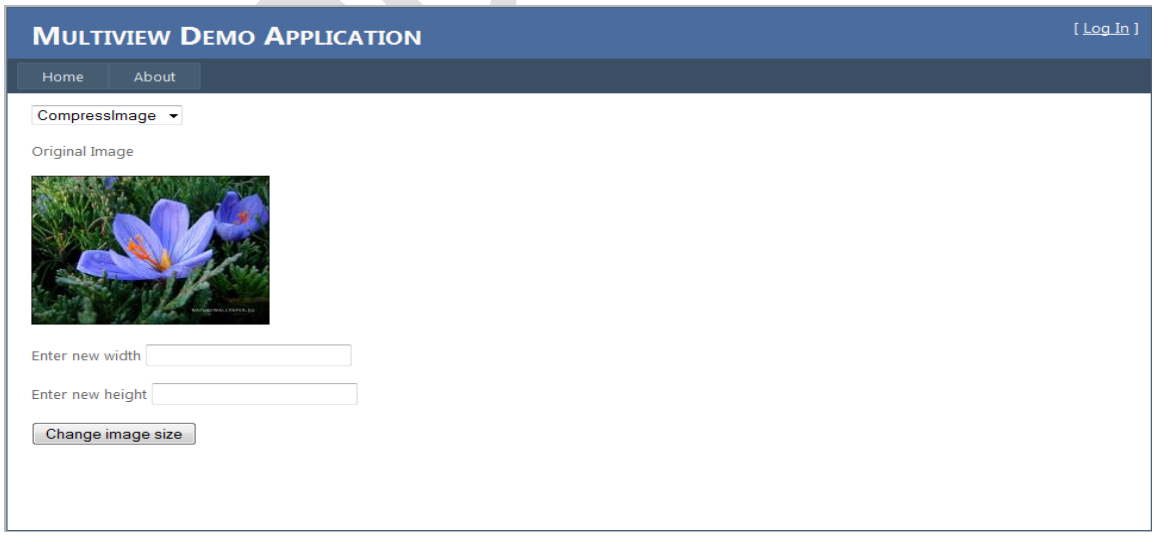
8. Handle SelectedIndexChanged event of DropDownList 'ddlType' to display the view based on the type selected from the DropDownList.
9. Handle the click event of the Upload button in 'UploadImage' view to save the images to the Image Path.Make use of File upload control.
10. Handle the click event of Button 'btnCreateThumbnail' in 'CompressImage' view to change the particular image width and height entered by the user.

Sample I/O:

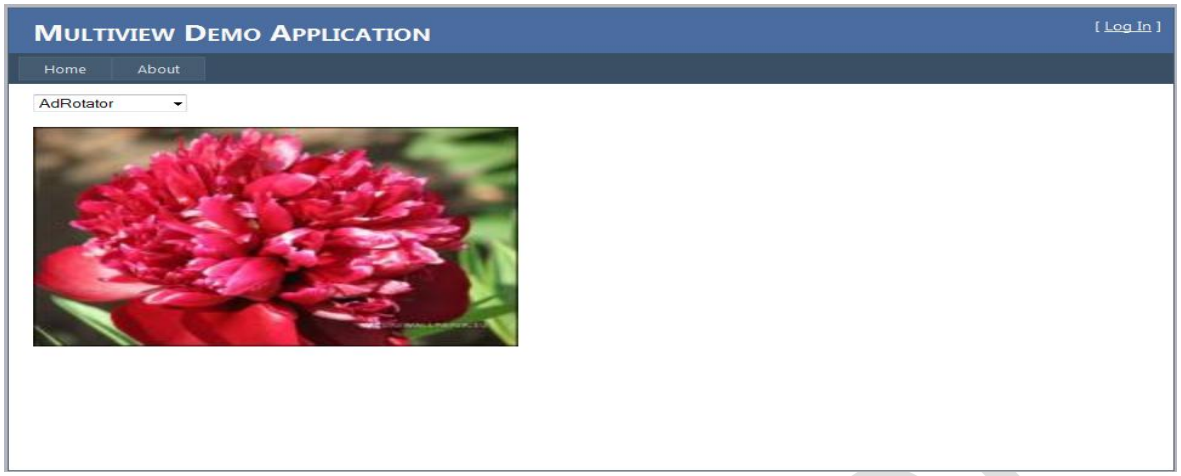
Screen for uploading the Image.



Screen for compressing the image.



Screen for rotating images.



Assignment 3:

Title: Create a web application using Wizard control. As the Wizard control is used to show the form in no of parts, create user registration form using this Wizard control save the details in the database and show all the details in another page.

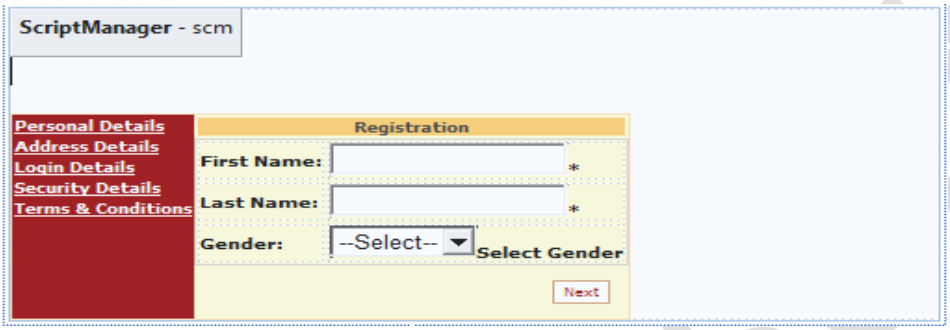
Description:

1. Create an Asp.net Web Site. Add a WebForm and name it as **Registration.aspx**
2. Create a new Database named as WizardDB and table UserDetails with the following columns:

Column Name	DataType	Constraints
PKUserId	int	Yes
FirstName	varchar(50)	No
LastName	varchar(50)	No
Gender	varchar(50)	No
Address	varchar(50)	No
State	varchar(50)	No
City	varchar(50)	No
Country	varchar(50)	No
ZipCode	int	No
MobileNumber	varchar(50)	No
EmailId	varchar(50)	No

Password	varchar(50)	No
SecurityQuestion	varchar(50)	No
SecurityAnswer	varchar(50)	No
IsActive	Bit	No

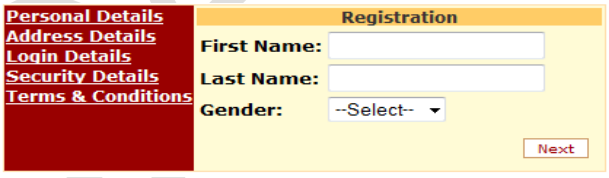
3. Design the form as follows:



4. Set the properties as shown below:

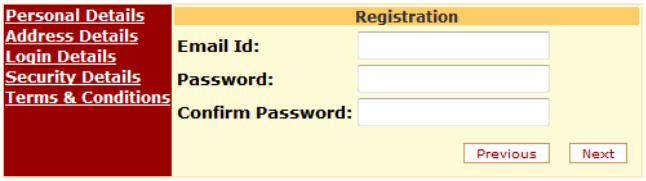
Control	Text	Name	Additional Properties
ScriptManager	scm		
Wizard		Wizard1	ActiveStepIndex=”0” HeaderText=”Registration”

5. Add the connection string in the web.Config.
6. Add 5 wizard steps 'wsPersonalInfo', 'wsAddDetails', 'wsLoginDetails', 'wsSecurity' and 'wsTerms' inside the wizard, so that whenever we click on next, the wizard step changes.
7. Drag and drop the controls in 'wsPersonalInfo' as displayed:

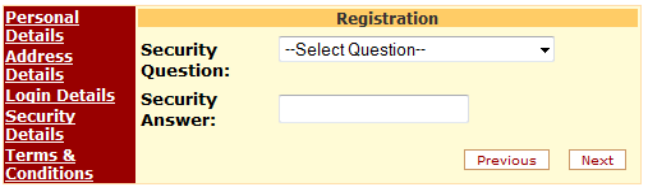


8. Drag and drop the controls in 'wsAddDetails' as displayed:

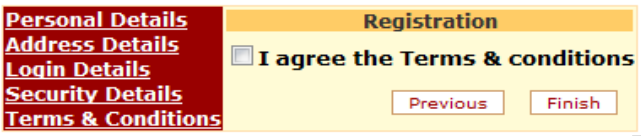
9. Drag and drop the controls in 'wsLoginDetails' as displayed:



10. Drag and drop the controls in 'wsSecurity' as displayed:



11. Drag and drop the controls in 'wsTerms' as displayed:



12. Create a class and name it as UserDetails.cs which contains a parameterized constructor and properties for each column in the table which is further used in the class 'UserDetailsBO' to get and set the values for UserDetails table.
13. Create a class and name it as 'UserDetailsBO'. Add the below code in the class which contains methods to insert the record and get the record per Id.
14. Handle the 'FinishButtonClick' event of the Wizard1 to redirect it to another page 'UserDetails.aspx' by passing the Id in url using QueryString.
15. Handle the active step changed event of the Wizard1 to insert data into the database by getting all the data information from all wizard steps and stores the Id(User Id got after inserting the record in database) in view state.
16. Create a new WebForm name it as 'UserDetails.aspx'. Drag and drop Labels to display the data (Name, Address, City, State, Country, Phone no) inserted in the database.
17. In the page load retrieve the Id from Query String and call the method 'getUserById' by passing Id. Display the required data.
18. Run the application.

Sample I/O:

After entering all the data in all Wizard Steps you can see the output as:

User Account

Welcome Meera Patel

Address Details:
Kharadi
Pune
Maharashtra
India
98745214565