ASP.NET MVC web apps – Patterns Summary – single entity

In the table cells below, briefly describe the important factors that implement the pattern. Then, use as a quick reference in the future. Revise when necessary.

| **Customer** | **Browser user action** | **View** | **View model** | **Controller** | **Manager** | **Map** | **Data store collection** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Get all** | Index or pressing back to list | Display a complete list from the database  Index.cshtml | Displays all fields from the base class  CustomerBase | Return the view calling the manager to get all function  Index() | Return the mapper from model to base and makes a connection to a database  IEnumerable<CustomerBase>CustomerGetAll() | Model to Base Class  m.Customer -> e.CustomerBase | dsEmployee  Gets model stored information |
| **Get one** | Clicking details | Display specific field information by ID  Details.cshtml | Reutilization of the base class  CustomerBase | Find the object, store it using a variable and if not null display it in the view.  Details(int? id) | Return Mapper from model to base and the specific variable containing the ID information  CustomerBase CustomerGetById(int id) | Model to Base Class. Details class if applicable  m.Customer -> e.CustomerBase | dsEmployee  Gets by id information |
| **Add new** | Clicking create | Displays empty fields to enter information  Create.cshtml | Uses an add class that uses but does not display the primary key  CustomerAdd | Check to make sure the info entered is valid or not null and then display details with the updated info  Create()  Create(CustomerAdd newItem) | Use a variable for the added item from the map and saves the changes to the Model.  CustomerBase  CustomerAdd(EmployeeAdd newItem) | From Add class to model  e.CustomerAdd -> m.Customer | dsEmployee  Insert data to database |
| **Edit existing** | Clicking edit | Displays fields that may be edited by the user  Edit.cshtml | Uses an edit contact info class that has limited access to change details  CustomerEditContactInfoForm &  CustomerEditContactInfo | Move from the Base class to the edit class with the object and return the view of with the editable fields.  Edit(int? id)  Edit?id , EmployeeEditContactInfo newItem) | Get the object and update it with the new values and save. Return the map from the model to class with the new object.  CustomerBase  CustomerEditContactInfo(CustomerEditContactInfo newItem) | From Base Class to edit contact info form  m.Customer -> e.CustomerBase | dsEmployee  Modify data on the persistent database |
| **Delete item** | Clicking delete | Displays the details and confirm removal  Delete.cshtml | Reutilization of the base class details to delete an entry | Return to index  Delete(int? id)  Delete(int?id, EmployeeAdd newItem) | Find the object, remove from the database with the corresponding model and save.  EmployeeDelete(int id) | Uses the Model to Base Class  - | dsEmployee  Removes data from the database |