**User Registration:**

**Controller**

[HttpGet]

public ActionResult RegisterUser()

{

Register register = new Register();

register.UserGroups = \_authendicationBL.GetUserTypeGroupBL();

register.UserTypes = \_authendicationBL.GetUserTypeBL();

List<string> marital = new List<string>();

marital.Add("Single");

marital.Add("Married");

register.Maritals = marital;

return View(register);

}

[HttpPost]

public ActionResult RegisterUser(Register register)

{

if (ModelState.IsValid)

{

if (\_authendicationBL.CheckUserNameBL(register.UserName))

{

ModelState.AddModelError(string.Empty, "Username already exist");

}

else

{

if (\_authendicationBL.RegisterUserBL(register))

{

return RedirectToAction("Login", "Account");

}

}

}

register.UserGroups = \_authendicationBL.GetUserTypeGroupBL();

register.UserTypes = \_authendicationBL.GetUserTypeBL();

List<string> marital = new List<string>();

marital.Add("Single");

marital.Add("Married");

register.Maritals = marital;

return View(register);

}

**Data Logic:**

public bool RegisterUser(Register register)

{

bool result = false;

try

{

User user = new User();

user.Active = true;

user.Age = Convert.ToInt32(register.Age);

user.CreatedAt = DateTime.Now.ToString();

user.DOB = register.DOB;

user.Gender = register.Gender;

user.Marital = register.Marital;

user.Name = register.Name;

user.UserId = Guid.NewGuid().ToString();

\_entity.Users.Add(user);

\_entity.SaveChanges();

MappingUserType mappingUserType = new MappingUserType();

mappingUserType.Active = true;

mappingUserType.UserId = user.UserId;

mappingUserType.UserTypeId = Convert.ToInt32(register.UserType);

\_entity.MappingUserTypes.Add(mappingUserType);

MappingUserGroup mappingUserGroup = new MappingUserGroup();

mappingUserGroup.Active = true;

mappingUserGroup.UserId = user.UserId;

mappingUserGroup.UserGroupId = Convert.ToInt32(register.UserGroupType);

\_entity.MappingUserGroups.Add(mappingUserGroup);

UserAuthentication userAuthentication = new UserAuthentication();

userAuthentication.Active = true;

userAuthentication.Password = register.Password;

userAuthentication.UserId = user.UserId;

userAuthentication.UserName = register.UserName;

\_entity.UserAuthentications.Add(userAuthentication);

\_entity.SaveChanges();

result = true;

}

catch

{

throw;

}

return result;

}

**Login**

[HttpGet]

public ActionResult Login()

{

return View();

}

[HttpPost]

public ActionResult Login(LoginUser login)

{

if (ModelState.IsValid)

{

if (login.UserName != "admin@scsd.com")

{

var userId = \_authendicationBL.CheckAuthenticationBL(login);

if (string.IsNullOrEmpty(userId))

{

ModelState.AddModelError(string.Empty, "Username or Password incorrect");

return View(login);

}

else

{

FormsAuthentication.SetAuthCookie(userId, false);

return RedirectToAction("UploadList", "Upload");

}

}

else

{

FormsAuthentication.SetAuthCookie("admin@scsd.com", false);

return RedirectToAction("UserList", "Admin");

}

}

return View(login);

}

**File Upload:**

**Controller**

[HttpGet]

public ActionResult UserUpload()

{

ViewBag.Entity = "UploadFile";

return View();

}

[HttpPost]

public ActionResult UserUpload(HttpPostedFileBase Baner, HttpPostedFileBase FileContent)

{

ViewBag.Entity = "UploadFile";

UploadFile uploadFile = new UploadFile();

uploadFile.Description = Request.Form["Description"] == null ? string.Empty : Request.Form["Description"].ToString();

if (Baner != null)

{

if (Baner.ContentLength != 0)

{

if (Path.GetExtension(Baner.FileName).ToLower() != ".jpg"

&& Path.GetExtension(Baner.FileName).ToLower() != ".png"

&& Path.GetExtension(Baner.FileName).ToLower() != ".gif"

&& Path.GetExtension(Baner.FileName).ToLower() != ".jpeg")

{

ModelState.AddModelError(string.Empty, "File cover must be image format");

return View(uploadFile);

}

uploadFile.Baner = CheckSumGenerator.GetByteFromStream(Baner.InputStream, Baner.ContentLength);

uploadFile.BanerApplicationType = Baner.ContentType;

}

}

if (FileContent != null)

{

if (FileContent.ContentLength != 0)

{

if (Path.GetExtension(FileContent.FileName).ToLower() != ".exe")

{

byte[] key1;

byte[] key2;

EllipticAsymmetric.KeyGenerator(out key1, out key2);

uploadFile.AsymKey = key2;

uploadFile.Type = EllipticAsymmetric.Encrypte(key1, Path.GetExtension(FileContent.FileName).ToLower());

uploadFile.Name = EllipticAsymmetric.Encrypte(key1, Path.GetFileNameWithoutExtension(FileContent.FileName).ToLower());

uploadFile.ApplicaitonType = EllipticAsymmetric.Encrypte(key1, FileContent.ContentType);

uploadFile.SymmeticKey = Guid.NewGuid().ToString();

byte[] fileData;

uploadFile.FileCheckSum = CheckSumGenerator.GetCheckSum(FileContent.InputStream, FileContent.ContentLength, out fileData);

if (\_UploadDataBL.CheckImageExistBL(uploadFile.FileCheckSum))

{

ModelState.AddModelError(string.Empty, "File already exist for your account");

return View(uploadFile);

}

else

{

uploadFile.FileContent = SymmetricEncryption.Encrypt(fileData, uploadFile.SymmeticKey);

uploadFile.FileId = Guid.NewGuid().ToString();

uploadFile.UserId = (User.Identity as SCSD.Web.SCSDIdentity).UserId;

if (\_UploadDataBL.InsertFileBL(uploadFile))

{

return RedirectToAction("UploadList", "Upload");

}

}

}

else

{

ModelState.AddModelError(string.Empty, "File format not suported.");

}

}

else

{

ModelState.AddModelError(string.Empty, "File must be upload");

return View(uploadFile);

}

}

else

{

ModelState.AddModelError(string.Empty, "File must be upload");

return View(uploadFile);

}

return View(uploadFile);

}

**Data Logic**

public bool InsertFile(UploadFile uploadFile)

{

try

{

FileMetadata fileMetaData = new FileMetadata();

fileMetaData.Active = true;

fileMetaData.ApplicationType = uploadFile.ApplicaitonType;

fileMetaData.Description = uploadFile.Description;

fileMetaData.Name = uploadFile.Name;

fileMetaData.Type = uploadFile.Type;

fileMetaData.Id = uploadFile.FileId;

\_entity.FileMetadatas.Add(fileMetaData);

\_entity.SaveChanges();

MappingFileCheckSum mappingFileChecksum = new MappingFileCheckSum();

mappingFileChecksum.FileCheckSum = uploadFile.FileCheckSum;

mappingFileChecksum.FileId = fileMetaData.Id;

mappingFileChecksum.Active = true;

\_entity.MappingFileCheckSums.Add(mappingFileChecksum);

MappingFileUser mappingFileUser = new MappingFileUser();

mappingFileUser.Active = true;

mappingFileUser.FileId = fileMetaData.Id;

mappingFileUser.UserId = uploadFile.UserId;

\_entity.MappingFileUsers.Add(mappingFileUser);

FileKey fileKey = new FileKey();

fileKey.Active = true;

fileKey.ASYMKey = uploadFile.AsymKey;

fileKey.SYMKey = uploadFile.SymmeticKey;

\_entity.FileKeys.Add(fileKey);

\_entity.SaveChanges();

MappingFileKey mappingFileKey = new MappingFileKey();

mappingFileKey.Active = true;

mappingFileKey.FileId = fileMetaData.Id;

mappingFileKey.KeyId = fileKey.Id;

\_entity.MappingFileKeys.Add(mappingFileKey);

FileBanar fileBanar = new FileBanar();

fileBanar.Active = true;

fileBanar.Banar = uploadFile.Baner;

fileBanar.Type = uploadFile.BanerApplicationType;

\_entity.FileBanars.Add(fileBanar);

\_entity.SaveChanges();

MappingFileBanar mappingFileBanar = new MappingFileBanar();

mappingFileBanar.Active = true;

mappingFileBanar.BanarId = fileBanar.Id;

mappingFileBanar.FileId = fileMetaData.Id;

\_entity.MappingFileBanars.Add(mappingFileBanar);

FileContent fileContent = new FileContent();

fileContent.Active = true;

fileContent.FileData = uploadFile.FileContent;

fileContent.FileId = fileMetaData.Id;

\_entity.FileContents.Add(fileContent);

\_entity.SaveChanges();

MappingFileContent mappingFileContent = new MappingFileContent();

mappingFileContent.Active = true;

mappingFileContent.ContentId = fileContent.Id;

mappingFileContent.FileId = fileMetaData.Id;

\_entity.MappingFileContents.Add(mappingFileContent);

\_entity.SaveChanges();

return true;

}

catch

{

throw;

}

}

**Admin:**

**Controller**

public class AdminController : Controller

{

private UploadDataBL \_UploadDataBL;

private AuthendicationBL \_authendicationBL = null;

public AdminController()

{

\_UploadDataBL = new UploadDataBL();

\_authendicationBL = new AuthendicationBL();

}

public ActionResult UserList()

{

ViewBag.Entity = "UserList";

List<UserDTO> users = new List<UserDTO>();

users = \_authendicationBL.GetUserIdentitysBL();

return View(users);

}

public ActionResult FileList()

{

ViewBag.Entity = "FileList";

List<UploadList> uploadList = new List<UploadList>();

uploadList = \_UploadDataBL.GetAllFileListBL();

return View(uploadList);

}

}