**Buchunt building and running instructions (docker):**

Install Docker + Docker compose

No IDE Required

docker compose up to build and run the project.

After changes are made to the code, you might need to rebuild the images.

docker compose down —rmi all -v will “clean up” the environment.

**Buchunt building and running instructions (native):**

Install an IDE capable of supporting C# ASP.NET projects such as Rider or Visual Studio

Open the solution in your IDE of choice and run the program.

When running natively, changes are applied immediately when rerunning. No need to worry about rebuilding images; although, Docker is the preferred method for deploying to a production environment.

**In Visual Studio:**

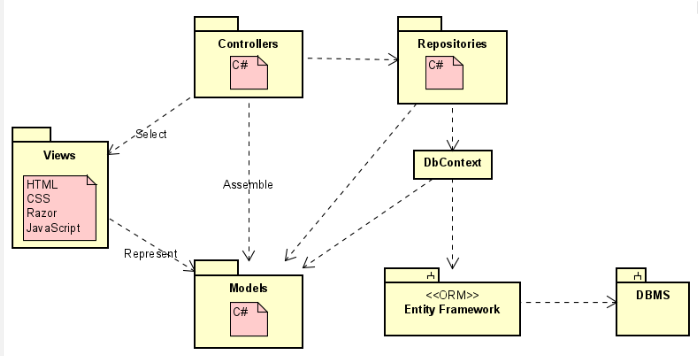
Open the .sln file. This is the whole project.

Click the outline of the play button to run the program with debug mode off.



**ASP.NET MVC Diagram**

This diagram shows how each of the files interact with each other when the application runs. The views hold the frontend code for the html. The controllers handle the backend logic that handles the HTTP requests from the client. The repositories communicate with the database to handle the SQL calls to select and insert records. For more information about the MVC design pattern, look at <https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?WT.mc_id=dotnet-35129-website&view=aspnetcore-7.0>.



**URL Path:**

https://localhost:/Controller/View Connected to Controller/Hunt ID

For example, to look at the details page for the Hunt controller with the id of 21. It would be:

<https://localhost/Hunt/Details/21>

The id is optional though, so pages like the index page in the Home controller is just this url:

<https://localhost/Home/Index>

**Admin Credentials:**

Email: [admin@admin.com](mailto:admin@admin.com)

Password: Scrum-daddy1

**Database:**

The database is sqlite. You can view the database by using DB Browser for SQLite or by using IDE plugins such as Rider’s database viewer.

Database tables are created through the use of DBContext classes. A migration is run and the database tables are generated.

**Create users via CSV (users controller/view):**

Uses Microsoft.Web.Helpers (NuGet)

Currently only functional on the front end. Back end code throwing errors when creating the user objects based on the CSV input.

[**https://learn.microsoft.com/en-us/aspnet/web-pages/overview/data/working-with-files**](https://learn.microsoft.com/en-us/aspnet/web-pages/overview/data/working-with-files)

**Email Server:**

The SMTP server is hosted on [elasticemail.com](https://elasticemail.com/). The account we made is a free trial that lets you have 100 free emails a day. Anymore than that and it will just refuse requests. In the project, it sends an email in the HuntController in the SendEmail method using the google account as the “from” email and uses the security token for the SMTP server.

If you need to change anything about this SMTP server, just login to the elasticemail account at elasticemail.com and go to settings. From there you can click the manage SMTP button to change anything.

**The elasticemail account info:**

* Email: [buchunt69@gmail.com](mailto:buchunt69@gmail.com)
* Pass: P@ss-w0rd!

**The google account account info:**

* Email: buchunt69@gmail.com
* Pass: P@ss-w0rd!