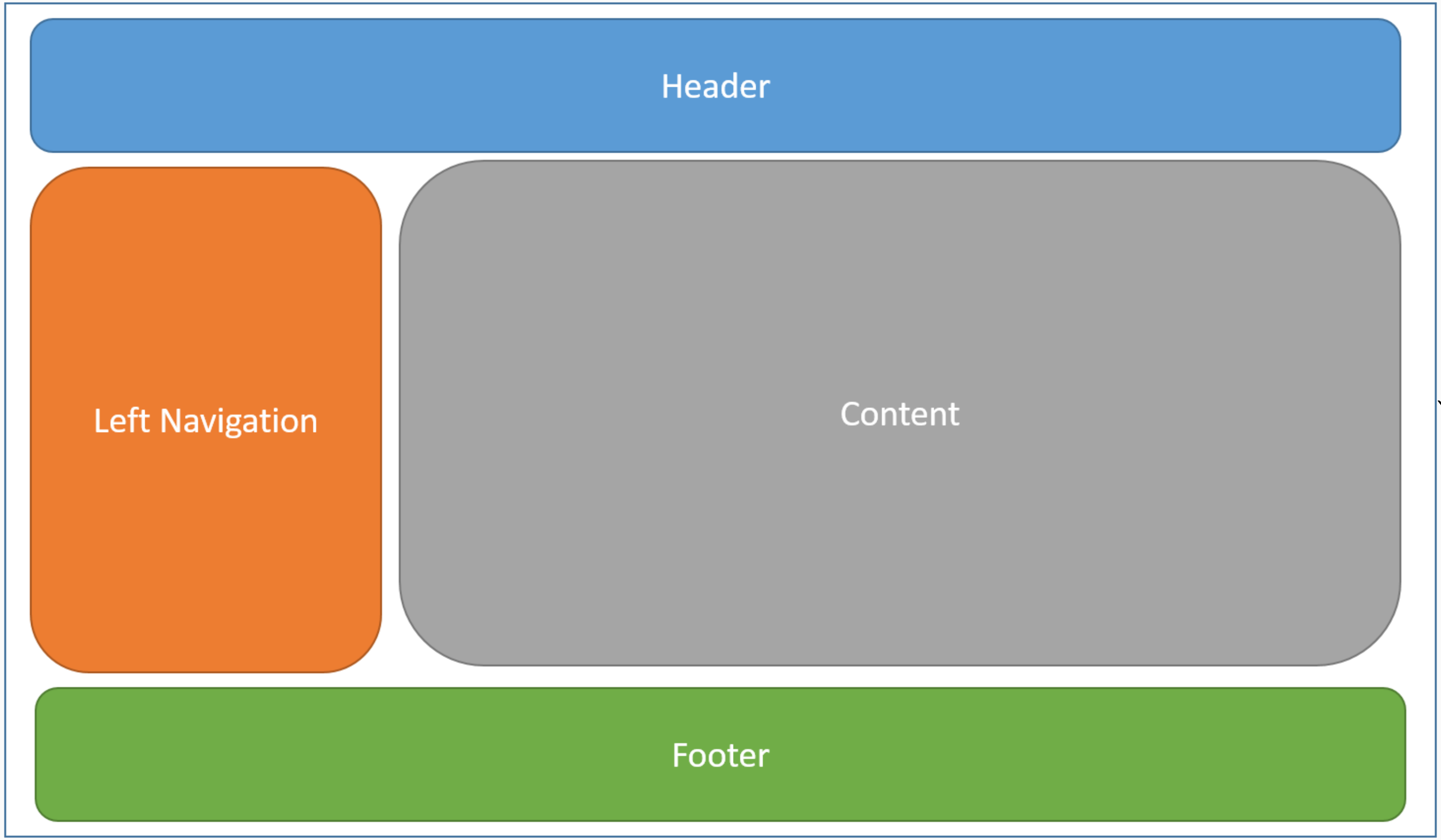
Școala
informală
de IT

Layout

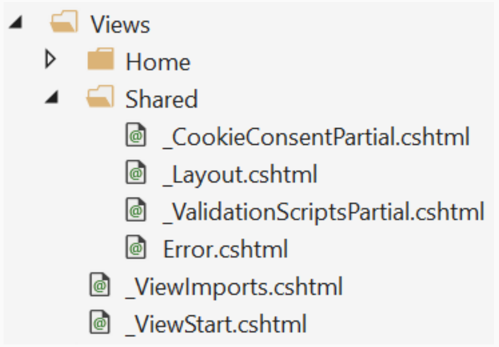
Layout



Layouts

- contains common user interface elements,
- defines a top level template for views in the app
- Provides a common look&feel for the user
- Can contain common styles, scripts for the application
- Can contain optional scripts, styles, html that appear only on certain pages
- Reduces code duplication in Views

- Apps can define more than one layout, with different views specifying different layouts



```
Views
  Home
  Shared
    _CookieConsentPartial.cshtml
    _Layout.cshtml
    _ValidationScriptsPartial.cshtml
    Error.cshtml
  _ViewImports.cshtml
  _ViewStart.cshtml
```

``

Specifying a layout

- Inside every view you can specify a layout or a different layout than the default
- Every View has a Layout property
- Every Layout file must specify `@RenderBody()`

```
@{
 Layout = "_Layout";
}
```

``

Sections

Sections

- A layout can specify one or more sections
- Provide a way where certain elements that come from certain views are injected into the layout
- These elements can be mandatory or optional.

```
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - FirstProject.MVC</title>

    <link rel="stylesheet“ href="~/lib/bootstrap/dist/css/bootstrap.css" />
    <link rel="stylesheet" href="~/css/site.css" />
    @RenderSection("Styles", required: false)
</head>
```

``

```
@section Styles{
    <link href="~/css/customer.css" rel="stylesheet" />
}
```

Sections

- A layout can specify one or more sections
- Provide a way where certain elements that come from certain views are injected into the layout
- These elements can be mandatory or optional.

```
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - FirstProject.MVC</title>

    <link rel="stylesheet“ href="~/lib/bootstrap/dist/css/bootstrap.css" />
    <link rel="stylesheet" href="~/css/site.css" />
    @RenderSection("Styles", required: false)
</head>
```

``

```
@section Styles{
    <link href="~/css/customer.css" rel="stylesheet" />
}
```

Şcoala informală de IT

Partial Views

Partial Views

A partial view is a Razor markup file (*.cshtml*) that renders HTML output *within* another markup file's rendered output.

- Break up large markup files into smaller components.
- Reduce the duplication of common markup content across markup files
- A partial view is a *.cshtml* markup file maintained within the *Views* folder (MVC)
- Partial view file names often begin with an underscore but is not required
- A partial View can be returned from an action by calling PartialView(), instead of View()
- Is discovered in the same way as Views
- Can be chained; A partial view can call another partial view
- You can use to display a complex object that is part of a model
- You can specify a name or a path for a partial view when you call it
- You can use it as HtmlHelper or TagHelper

Partial Views

```
@model ReadRPModel

<h2>@Model.Article.Title</h2>
@* Pass the author's name to Pages\Shared\_AuthorPartialRP.cshtml *@
@await Html.PartialAsync("../Shared/_AuthorPartialRP", Model.Article.AuthorName)
@Model.Article.PublicationDate

}
```

Partial Views

```
<partial name="Shared/_OrderPartial.cshtml" for="Order">
```

```
@foreach (var order in Model.Orders)
{
    @await Html.Partial("_OrderPartial", order)
}
```