



Faculty of Electronics
and Information
Technology

WARSAW UNIVERSITY OF TECHNOLOGY

Graphical User Interfaces (EGUI)

MVC

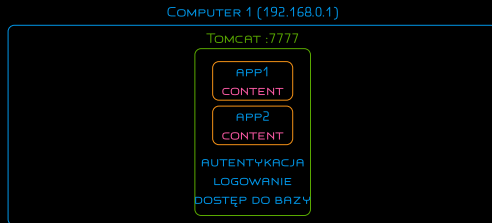
Julian Myrcha
Institute of Computer Science
October 22, 2024





Servers and Microservices

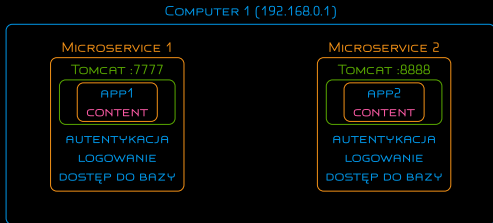
- The standard approach is monolithic servers hosting multiple applications
- traditional approach
- in one place user management and environment configuration





Servers and Microservices

- The standard approach is monolithic servers hosting multiple applications
- The approach of microservices is gaining popularity - separate programs that implement parts of the functionality independently
- breakdown into smaller parts makes testing easier
- libraries and tools to facilitate configuration





Servers and Microservices

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

**Servers and
Microservices**

Web server

Forms (1)

Devtools

server side

Javascript

HTTP

HTML

document
structure

CSS

Bootstrap

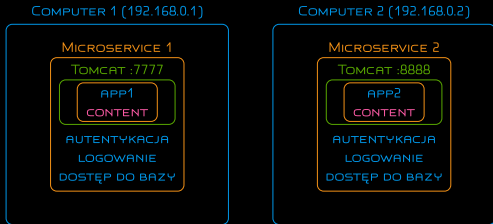
Character
encoding

MVC

Summary

State

- The standard approach is monolithic servers hosting multiple applications
- The approach of microservices is gaining popularity - separate programs that implement parts of the functionality independently
- breakdown into smaller parts makes testing easier
- libraries and tools to facilitate configuration
- you can easily create distributed solutions





Web server

- browser check what has been sent and use suitable display method
 - plik html** html - render graphics content generated from description
 - plik tekstowy** text - render plain text

```
1 jmy@bardex:$ telnet localhost 8080
2 Trying 127.0.0.1...
3 Connected to localhost.
4 Escape character is '^'.
5 GET /FirstWebApp/first HTTP/1.1
6 HOST: localhost
7
8 HTTP/1.1 200 OK
9 Server: GlassFish Server Open Source Edition 4.1.1
10 X-Powered-By: Servlet/3.1 JSP/2.3 (GlassFish Server Open Source Edition 4.1.1 Java/Oracle
   ↳ Corporation/1.8)
11 Date: Fri, 21 Oct 2016 14:39:03 GMT
12 Content-Length: 23
13
14 Served at: /FirstWebAppConnection closed by foreign host.
15 jmy@bardex:$
```

Web server

- browser check what has been sent and use suitable display method
 - plik html** html - render graphics content generated from description
 - plik tekstowy** text - render plain text

```
1 curl http://galera.ii.pw.edu.pl/index.html
2 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
3 >
4 <HTML>
5 <HEAD>
6   <META http-equiv=Content-Type content="text/html; charset=iso-8859-2">
7   <link rel="stylesheet" type="text/css" href="/galera.css">
8   <TITLE>Serwer galera.ii.pw.edu.pl</TITLE>
9 </HEAD>
10
11 <BODY>
12 <CENTER>
13 ...
14 </HTML>
```



Forms (1)

Example of the page with HTML form:

```
1  <!DOCTYPE html>
2  <html>
3  <head><meta charset="UTF-8">
4      <title>EGUI</title>
5  </head>
6  <body>
7  <h2>Formy</h2>
8
9  <form action="first">
10     Imie:<br>
11     <input type="text" name="firstname" value="Johny">
12     <br>
13     Nazwisko:<br>
14     <input type="text" name="lastname" value="Bean">
15     <br><br>
16     <input type="submit" name = 'btn' value="Send">
17     <input type="submit" name = 'btn' value="Cancel">
18 </form>
19
20 <p>if you press button "Send", data will be sent to page "first"</p>
21 </body>
22 </html>
```



Forms (2)

Server gets following data:

firstname -Johny

lastname -Bean

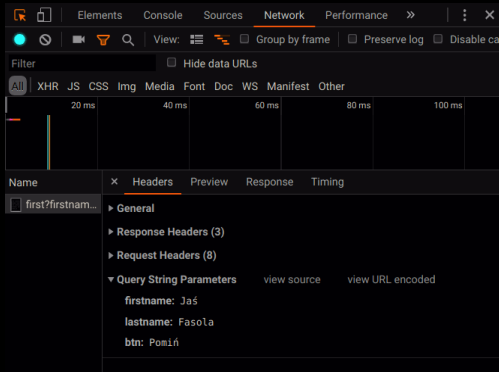
btn -Send

For submit elements only one button produces data!



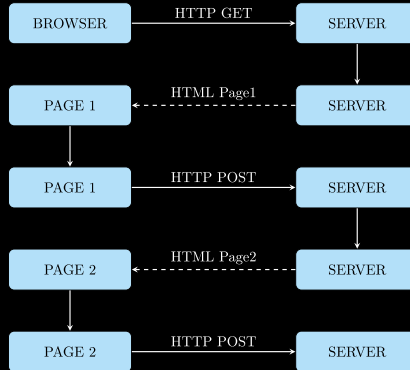
Devtools

how to see form data: (Ctrl-Shift-C w chrome)





Interaction between server and browser for normal web app





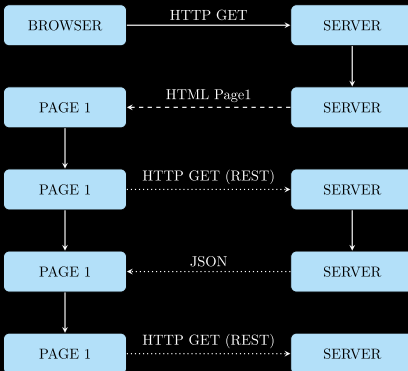
Javascript

```
1  <!DOCTYPE html>
2  <html>
3  <body>
4
5  <h2>My First JavaScript</h2>
6
7  <button type="button"
8  onclick="document.getElementById('demo').innerHTML = Date()">
9  Click me to display Date and Time.</button>
10
11 <p id="demo"></p>
12
13 </body>
14 </html>
```





Javascript





HTTP Protocol

HTTP 1.0 - GET, POST, HEAD

HTTP 1.1 - GET, POST, HEAD, PUT, OPTIONS, DELETE, TRACE,
CONNECT

GET

```
1 GET /servlet/HelloServlet?userid = john HTTP/1.0
```

POST

```
1 POST /servlet/helloServlet HTTP/1.0
2 User-Agent: MOZILLA/1.0
3 Content-Type: application/x-www-form-urlencoded
4 ContentLength: 11
5 userid=john
```

one could use **curl** or browser

```
1 curl www.google.com
```



HTTP Protocol - response

Response class codes:

- 1xx** 1xx - Information codes
- 2xx** 2xx - Confirmation codes
- 3xx** 3xx - Redirection codes
- 4xx** 4xx - Application error codes
- 5xx** 5xx - Internal error codes



HTTP Protocol - response

description	code	meaning
200	OK	Content of the requested document
400	Bad Request	Invalid request
401	Unauthorized	Unauthorized request
404	Not Found	Resource with given url was not found on the server
408	Request Timeout	Timeout
500	Internal Server Error	Internal Server Error
503	Service Unavailable	Service Unavailable



Historia

- HTML has its roots in Standard General Markup Language, or SGML



Historia

- HTML has its roots in Standard General Markup Language, or SGML
 - Language to describe language



Historia

- HTML has its roots in Standard General Markup Language, or SGML
 - Language to describe language
- **HTML, or HyperText Markup Language, began as an application of SGML**



Historia

- HTML has its roots in Standard General Markup Language, or SGML
 - Language to describe language
- HTML, or HyperText Markup Language, began as an application of SGML
 - **Hypertext simply means that the text contains links to other texts**



Historia (cd)

HTML1 HTML1 - lack of tags: **img** or **table**





Historia (cd)

HTML1 HTML1 - lack of tags: **img** or **table**

HTML 2.0 HTML 2.0 - little changes



Historia (cd)

HTML1 HTML1 - lack of tags: **img** or **table**

HTML 2.0 HTML 2.0 - little changes

HTML 3.2 - style sheets, tables, interaction with java language (applets), fonts



Historia (cd)

HTML1 HTML1 - lack of tags: **img** or **table**

HTML 2.0 HTML 2.0 - little changes

HTML 3.2 - style sheets, tables, interaction with java language (applets), fonts

HTML 4.0 - ability to set a document type:

Transitional - ability to intermix HTML 3.2 and HTML 4

Strict - only HTML4 allowed

Framestet - a lot of documents can be combined into frames



Historia (cd)

HTML1 HTML1 - lack of tags: **img** or **table**

HTML 2.0 HTML 2.0 - little changes

HTML 3.2 - style sheets, tables, interaction with java language (applets), fonts

HTML 4.0 - ability to set a document type:

Transitional - ability to intermix HTML 3.2 and HTML 4

Strict - only HTML4 allowed

Framestet - a lot of documents can be combined into frames

HTML 4.0 - lack of definition how to parse HTML page



Historia (cd)

HTML1 HTML1 - lack of tags: **img** or **table**

HTML 2.0 HTML 2.0 - little changes

HTML 3.2 - style sheets, tables, interaction with java language (applets), fonts

HTML 4.0 - ability to set a document type:

Transitional - ability to intermix HTML 3.2 and HTML 4

Strict - only HTML4 allowed

Framestet - a lot of documents can be combined into frames

HTML 4.0 - lack of definition how to parse HTML page

XHTML 1.0 • require tags in lower letters

- require closing tags for not empty elements (**/>**)
- uses only **application/xml+xml** MIME type
- there was a problem with browser support



History

- adding dOM and JavaScript (Netscape Navigator 2.0, 1996)
- adding applets and plugins
 - Adobe Flash
 - Shockwave
 - Aplety Javy
 - Real Player
 - Apple QuickTime
- which introduced a lot of security issues ...



HTML5

- HTML parsing and creation of DOM even if document is not formed properly



HTML5

- HTML parsing and creation of DOM even if document is not formed properly
- adds new elements to support multimedia and web applications



HTML5

- HTML parsing and creation of DOM even if document is not formed properly
- adds new elements to support multimedia and web applications



HTML5

- HTML parsing and creation of DOM even if document is not formed properly
- adds new elements to support multimedia and web applications
- **redefines the rules and semantics of existing HTML elements**



HTML5

- HTML parsing and creation of DOM even if document is not formed properly
- adds new elements to support multimedia and web applications
- redefines the rules and semantics of existing HTML elements
- **eliminuje potrzebę pluginów (bo implementuje to co one dostarczały)**



HTML5

- HTML parsing and creation of DOM even if document is not formed properly
- adds new elements to support multimedia and web applications
- redefines the rules and semantics of existing HTML elements
- eliminuje potrzebę pluginów (bo implementuje to co one dostarczały)
- **wprowadza funkcje które definiował XHTML**



HTML5

- HTML parsing and creation of DOM even if document is not formed properly
- adds new elements to support multimedia and web applications
- redefines the rules and semantics of existing HTML elements
- eliminuje potrzebę pluginów (bo implementuje to co one dostarczały)
- wprowadza funkcje które definiował XHTML

WHATWG - Web Hypertext Application Technology Working Group

- żywy dokument - zmiany wprowadzane na bieżąco

W3C - World Wide Web Consortium

- wersjonowane - "Working Draft" -> "Candidate Recommendation" -> "W3C Recommendation"



HTML5 - przykład

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>PSI</title>
5   </head>
6   <body>
7     <p>Wykład PSI</p>
8   </body>
9 </html>
```

- DOCTYPE - wymagane, bo bez tego browser nie wie, że to HTML5
- jak nie ma to browser może wybrać to co chce (np. zgodność ze swoimi starymi błędami)

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
2 "http://www.w3.org/TR/html4/strict.dtd">
```



HTML5 - tryb HTML (1)

If it's served as text/html, the following rules apply:

- The document must include an HTML5 DOCTYPE.
- Start tags are not required for every element.
- End tags are not required for every element.
- Only void elements such as br, img, and link may be "self-closed" with />.
- Tags and attributes are case-insensitive.
- Attributes do not need to be quoted.
- Some attributes may be empty (such as checked and disabled).
- Special characters, or entities, do not have to be escaped.



HTML5 - tryb XHTML (1)

- XHTML 1.0 was "a reformulation of HTML 4 as an XML 1.0

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" />
5   <title>Lena</title>
6 </head>
7 <body>
8   <p>
9     
10    najważniejsze zdjęcie nauki
11   </p>
12   <script src="jakis-skrypt.js" />
13 </body>
14 </html>
```



HTML5 - tryb XHTML (1)

- XHTML 1.0 was "a reformulation of HTML 4 as an XML 1.0
- (X)HTML5 that's written and parsed using the syntax rules of XML and served with a Content-type: application/xml+xml

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" />
5   <title>Lena</title>
6 </head>
7 <body>
8   <p>
9     
10    najważniejsze zdjęcie nauki
11   </p>
12   <script src="jakis-skrypt.js" />
13 </body>
14 </html>
```



HTML5 - tryb XHTML (1)

- XHTML 1.0 was "a reformulation of HTML 4 as an XML 1.0
- (X)HTML5 that's written and parsed using the syntax rules of XML and served with a Content-type: application/xml+xml
- **wymaga xmlns - XML name space**

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" />
5   <title>Lena</title>
6 </head>
7 <body>
8   <p>
9     
10    najważniejsze zdjęcie nauki
11   </p>
12   <script src="jakis-skrypt.js" />
13 </body>
14 </html>
```



HTML5 - tryb XHTML (1)

- XHTML 1.0 was "a reformulation of HTML 4 as an XML 1.0
- (X)HTML5 that's written and parsed using the syntax rules of XML and served with a Content-type: application/xml+xml
- wymaga xmlns - XML name space
- **serwer webowy musi wiedzieć że wysyła xhtml (informacja Content-type: application/xml+xml w nagłówku komunikatu). Najczęściej robi gdy dokument ma nazwę *.xhtml**

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" />
5   <title>Lena</title>
6 </head>
7 <body>
8   <p>
9     
10    najważniejsze zdjęcie nauki
11  </p>
12  <script src="jakis-skrypt.js" />
13 </body>
14 </html>
```



HTML5 - tryb XHTML (2)

- Non-void elements with a start tag must have an end tag (p and li, for example).

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" /> <title>Lena</title>
5 </head>
6 <body>
7   <p>
8 </p>
9 </body>
10 </html>
```




HTML5 - tryb XHTML (2)

- Non-void elements with a start tag must have an end tag (p and li, for example).
- Any element may be "self-closed" using />.

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" /> <title>Lena</title>
5 </head>
6 <body>
7   <p>
8 </p>
9 </body>
10 </html>
```



HTML5 - tryb XHTML (2)

- Non-void elements with a start tag must have an end tag (p and li, for example).
- Any element may be "self-closed" using />.
- Tags and attributes are case sensitive, typically lowercase.

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" /> <title>Lena</title>
5 </head>
6 <body>
7   <p>
8 </p>
9 </body>
10 </html>
```



HTML5 - tryb XHTML (2)

- Non-void elements with a start tag must have an end tag (p and li, for example).
- Any element may be "self-closed" using />.
- Tags and attributes are case sensitive, typically lowercase.
- **Attribute values must be enclosed in quotes.**

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" /> <title>Lena</title>
5 </head>
6 <body>
7   <p>
8 </p>
9 </body>
10 </html>
```



HTML5 - tryb XHTML (2)

- Non-void elements with a start tag must have an end tag (p and li, for example).
- Any element may be "self-closed" using />.
- Tags and attributes are case sensitive, typically lowercase.
- Attribute values must be enclosed in quotes.
- Empty attributes are forbidden (checked must instead be checked="checked" or checked="true").

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" /> <title>Lena</title>
5 </head>
6 <body>
7   <p>
8 </p>
9 </body>
10 </html>
```



HTML5 - tryb XHTML (2)

- Non-void elements with a start tag must have an end tag (p and li, for example).
- Any element may be "self-closed" using />.
- Tags and attributes are case sensitive, typically lowercase.
- Attribute values must be enclosed in quotes.
- Empty attributes are forbidden (checked must instead be checked="checked" or checked="true").
- **Special characters must be escaped using character entities.**

```
1 <!DOCTYPE html>
2 <html xmlns="http://www.w3.org/1999/xhtml">
3 <head>
4   <meta charset="utf-8" /> <title>Lena</title>
5 </head>
6 <body>
7   <p>
8 </p>
9 </body>
10 </html>
```



struktura

- kiedyś używaliśmy taga div (i określaliśmy jego położenie)
- w HTML5 mamy tagi: **header**, **av** i **ooter**

```
1 <header>
2 <h1>HTML5 <i>0 strukturze</i></h1>
3 <h2>Wszystkie działy</h2>
4 <nav>
5     <ul>
6         <li><a href="#">Dalej</a></li>
7         <li><a href="#">Na tej</a></li>
8         <li><a href="#">Wykłady & Cwiczenia</a></li>
9     </ul>
10 </nav>
11 </header>
```





Shortest possible html introduction (1)

- <http://www.w3schools.com/html/>

call from command line:

```
curl http://localhost:8080/FirstWebApp/
```



Shortest possible html introduction (2)

- structure of the file:
 - optional header (`<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">`)
 - tag `<html>` containing one `<head>` and one `<body>`)

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
2 <html>
3   <head>
4     <meta http-equiv="content-type" content="text/html; charset=UTF-8" >
5     <title>...</title>
6   </head>
7   <body>
8     ...
9   </body>
10 </html>
```




Shortest possible html introduction (3)

text is copied to the output (bringing into account current formatting) but:

- many spaces reduced to one
- newline character replaced by space
- tabular character replaced by space

structure of the html tag:

example	description
<code>
</code>	self ending tag
<code><p> something </p></code>	beginning and ending tag surrounding some html
<code><li style="margin: 0 0 15px 0;"></code>	tag with attributes

Shortest possible html introduction (4)

- tables often used for formatting the page:

```
1 <table>
2 <thead>
3   <tr>
4     <td>Konsultacje:</td>
5     <td>Friday, 14-16; room. 308A</td>
6   </tr>
7 </thead>
8 <tbody>
9   <tr>
10    <td>Telefon (PW):</td>
11    <td>(22) 234-5413</td>
12  </tr>
13  <tr>
14    <td>jmy@ii.pw.edu.pl</td>
15  </tr>
16 </tbody>
17 </table>
```



Shortest possible html introduction (5)

```
1 Konsultacje:      Friday, 14-16; room. 308A
2 Telefon (PW):      (22) 234-5413
3 jmy@ii.pw.edu.pl
```

- we got columnar layout
- `<thead>` and `<tbody>` are optional

Concatenation of the cells:

`atribut colspan="2"` attribute `colspan="2"` - cell occupy 2 columns
`atribut rowspan="2"` attribute `rowspan="2"` - cell occupy 2 rows



A few words about CSS (1)

- Separation between view structure and visual attributes
- Supports reuse (if separate css files used)
- loading position starts with html file location

```
1 <head>
2 <link href="css/styles.css"
3   rel="stylesheet" type="text/css"/>
4 ...
5 </head>
```



A few words about CSS (2)

- We can define css file in the html file

```
1 <head>
2 <style type="text/css">
3   td { color: red; }
4   .btn { font-family: Courier; }
5 </style>
6 </head>
```

- or even on the tag level

```
1 <div style="color: blue; background-color: black"> ...
```



A few words about CSS (3)

- if many definitions for the same item the last is the winner (cascading ...)
- For each tag there is a lookup for all applying styles

po typie taga type of the tag

po klasie dodanej tagowi class of the tag

po unikalnym identyfikatorze taga id of the tag

```
1 h1 {font-family: Courier; }
```

- Means that all tag headers on h1 level will be in Courier font

```
1 .btn {font-family: Courier; }
```

- Means that all tags with class btn will be in font Courier **<p class="btn"/p> ... </p>**



Selectors

- by id:

```
1 #balbinka { ...  
2 <h1 id="balbinka">
```

- by nesting level:

```
1 td div { } ustawienia dla div ktore jest w td (dowolny poziom)  
2 <td> ... <div> ... </div> .. <td>
```

- by ndirect nesting level:

```
1 td > div { } ustawienia dla div ktore jest w td (bezposrednio)  
2 <td> <div> ... </div> <td>
```

- many settings for the seme selector:

```
1 h1, h2 {}
```

- tag with class:

```
1 td.btn {}  
2 <td class="btn">
```



common attributes:

- Margin
- Border
- Padding
- Content



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <style>
5   table, th, td {
6     border: 1px solid black;
7   }
8 </style>
9 </head>
10 <body>
11 ...
12 </body>
```




What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development



What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins



What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive designs - Bootstrap's responsive CSS adjusts to phones, tablets, and desktops



What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive designs - Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
- Browser compatibility: Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)



What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive designs - Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
- Browser compatibility: Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)

- **Mamy 2 wersje biblioteki:**

Bootstrap 3 - Bootstrap 3 is the most stable version of Bootstrap, and it is still supported by the team for critical bugfixes and documentation changes



What is Bootstrap?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive designs - Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
- Browser compatibility: Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)

- **Mamy 2 wersje biblioteki:**

Bootstrap 3 - Bootstrap 3 is the most stable version of Bootstrap, and it is still supported by the team for critical bugfixes and documentation changes

Bootstrap 4 - Bootstrap 4 is the newest version of Bootstrap; with new components, faster stylesheet and more responsiveness



Where to Get Bootstrap 4?

- Instalacja przez npm-a

```
1 npm install bootstrap
```





Where to Get Bootstrap 4?

- Instalacja przez npm-a
- Include Bootstrap 4 from a CDN

```
1 <!-- CSS only -->
2 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
  ↳ integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYXxFfc+NcPb1dKGj7Sk"
  ↳ crossorigin="anonymous">
3
4 <!-- JS, Popper.js, and jQuery -->
5 <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
  ↳ integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
  ↳ crossorigin="anonymous"></script>
6 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
  ↳ integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
  ↳ crossorigin="anonymous"></script>
7 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
  ↳ integrity="sha384-0gVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI"
  ↳ crossorigin="anonymous"></script>
```




Where to Get Bootstrap 4?

- Instalacja przez npm-a
- Include Bootstrap 4 from a CDN
- Download Bootstrap 4 from getbootstrap.com



Where to Get Bootstrap 4?

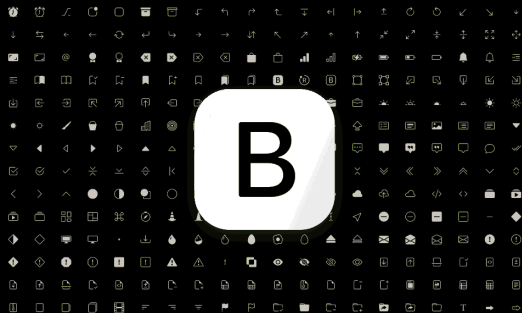
- Instalacja przez npm-a
- Include Bootstrap 4 from a CDN
- Download Bootstrap 4 from getbootstrap.com
- Bootstrap requires the use of the HTML5 doctype
- Bootstrap is developed mobile first, a strategy in which we optimize code for mobile devices first and then scale up components as necessary using CSS media queries.

```
1 <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```



Bootstrap Icons

Bootstrap has its own open source SVG icon library





Starter template

```
1  <!doctype html>
2  <html lang="en">
3    <head>
4      <!-- Required meta tags -->
5      <meta charset="utf-8">
6      <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
7
8      <!-- Bootstrap CSS -->
9      <link rel="stylesheet"
10        ↪ href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
11        ↪ integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYxXfFc+NcPb1dKGj7Sk"
12        ↪ crossorigin="anonymous">
13
14      <title>Hello, world!</title>
15    </head>
16    <body>
17      <h1>Hello, world!</h1>
18
19      <!-- Optional JavaScript -->
20      <!-- jQuery first, then Popper.js, then Bootstrap JS -->
21      <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"
22        ↪ integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"
23        ↪ crossorigin="anonymous"></script>
24      <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
25        ↪ integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
26        ↪ crossorigin="anonymous"></script>
27      <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
28        ↪ integrity="sha384-0gVRvuATP1z7JjHLku0U7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JIKI"
29        ↪ crossorigin="anonymous"></script>
30    </body>
31  </html>
```



Formy

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <title>Bootstrap Example</title>
5    <meta charset="utf-8">
6    <meta name="viewport" content="width=device-width, initial-scale=1">
7    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css">
8    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
9    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.16.0/umd/popper.min.js"></script>
10   <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"></script>
11 </head>
12 <body>
13
14 <div class="container">
15   <h2>Form Validation</h2>
16   <p>In this example, we use <code>.was-validated</code> to indicate what's missing before submitting
17   ↪ the form:</p>
18   <form action="/action_page.php" class="was-validated">
19     <div class="form-group">
20       <label for="uname">Username:</label>
21       <input type="text" class="form-control" id="uname" placeholder="Enter username" name="uname"
22       ↪ required>
23       <div class="valid-feedback">Valid.</div>
24       <div class="invalid-feedback">Please fill out this field.</div>
25     </div>
26     <div class="form-group">
27       <label for="pwd">Password:</label>
28       <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd"
29       ↪ required>
30       <div class="valid-feedback">Valid.</div>
31       <div class="invalid-feedback">Please fill out this field.</div>
32     </div>
33     <div class="form-group form-check">
```



Formy

```
31     <label class="form-check-label">
32         <input class="form-check-input" type="checkbox" name="remember" required> I agree on blabla.
33         <div class="valid-feedback">Valid.</div>
34         <div class="invalid-feedback">Check this checkbox to continue.</div>
35     </label>
36 </div>
37 <button type="submit" class="btn btn-primary">Submit</button>
38 </form>
39 </div>
40
41 </body>
42 </html>
```



Form Validation

In this example, we use `.was-validated` to indicate what's missing before submitting the form:

Username:

 ✓

Valid.

Password:

 ✓

Valid.

☐ I agree on blabla.

[Check this checkbox to continue.](#)

Submit



Coding characters

CP852																
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	Znaki kontrolne															
1x																
2x	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{		}	~	ZK
8x	Ç	ü	é	â	ä	û	ć	ç	ł	ë	Ö	ö	î	Ż	Ä	Ć
9x	É	Í	Í	ô	ö	Ł	ł	Ś	ś	Ö	Ü	Ť	ť	Ł	×	č
Ax	á	í	ó	ú	À	à	Ž	ž	Ę	ę	¬	ž	Č	š	«	»
Bx	☒	☒	☒		†	Á	Â	Ě	Š	‡		¶	¶	Ž	ž	‡
Cx	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł	Ł
Dx	đ	Đ	Đ	Ě	đ	Ň	í	î	ě	Ĵ	ŕ	■	■	Ť	Ů	■
Ex	Ó	ß	Ô	Ň	ń	ň	Š	š	Ř	Ú	ř	Ů	ý	Ý	ť	'
Fx	SHY	"	„	”	„	§	÷	„	°	“	•	ü	Ř	ř	■	NBSP



Coding characters

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

Kody znaków

Wyświetlanie UTF
Błędy związane z
kodowaniem

MVC

Summary

State

ISO/IEC 8859-2:1999																
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	Znaki kontrolne															
1x																
2x	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6x	'	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{		}	~	
8x	Nieużywane															
9x																
Ax	NBSP	Ą	~	Ł	ł	Ł	Ś	ś	ˆ	Š	š	Ť	Ž	SHY	Ž	Ž
Bx	°	ą	ˆ	ł	ł	ł	ś	ś	ˆ	š	š	ť	ž	ˆ	ž	ž
Cx	Ř	Á	Â	Ã	Ä	Å	Ć	Ç	Č	É	Ę	Ě	Ě	Í	Î	Ď
Dx	Đ	Ń	Ň	Ó	Ô	Õ	Ö	×	Ř	Ú	Ú	Ů	Ů	Ý	Ť	ß
Ex	ř	á	â	ã	ä	å	ć	ç	č	é	ę	ě	ě	í	î	ď
Fx	đ	ñ	ň	ó	ô	õ	ö	+	ř	ú	ú	ů	ů	ý	ť	·



Coding characters

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

Kody znaków

Wyświetlanie UTF
Błędy związane z
kodowaniem

MVC

Summary

State

Windows-1250																
	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL
8x	€	NZ	,	NZ	"	...	†	‡	NZ	%	Š	‹	Š	Ť	Ž	Ž
9x	NZ	ˆ	ˆ	"	"	•	—	—	NZ	™	š	›	š	ř	ž	ž
Ax	NBSP	ˆ	ˆ	Ł	□	Ą	ı	§	"	©	§	«	¬	SHY	®	Ž
Bx	°	±	„	ı	ˆ	µ	¶	·	„	ą	§	»	Ł	ˆ	ř	ž
Cx	Ř	Á	Â	Ã	Ä	Í	Ó	Ç	Č	É	Ê	Ë	Ë	Í	Î	Ď
Dx	Đ	Ñ	Ñ	Ó	Ô	Õ	Ö	×	Ř	Ú	Ú	Ů	Ů	Ý	Ť	ß
Ex	í	á	â	ã	ä	í	ć	ç	č	é	ę	ë	ë	í	î	ď
Fx	đ	ñ	ñ	ó	ô	õ	ö	+	ř	ú	ú	ů	ů	ý	ť	·





Printing codes (1/4)

```
1 import java.io.UnsupportedEncodingException;
2 import java.nio.charset.Charset;
3 import java.util.ArrayList;
4 import java.util.List;
5
6 public class CharacterEncodings {
7     static String convert(String input, String encoding)
8         throws UnsupportedEncodingException {
9         byte[] encoded_input = Charset.forName(encoding).encode(input).array();
10        List<String> res = new ArrayList<String>();
11        for(int i=0; i<encoded_input.length; i++)
12            res.add(String.format("%02X", encoded_input[i]));
13        return String.join(":", res);
14    }
15    public static void main(String[] args) throws UnsupportedEncodingException {
16        System.out.println(Charset.defaultCharset().displayName()); // UTF-8
17        System.out.println(Byte.SIZE); // 8
18        System.out.println(Character.SIZE); // 16
19        System.out.println(Double.SIZE); // 64
20        System.out.println("A = "+convert("A", "UTF-8")); // A = 41
21        System.out.println("A = "+convert("A", "UTF-8")); // A = C4:84:00
22    }
23 }
```

Servlet Response in UTF-8 (2/4)

- plik java - w UTF-8
- nagłówek HTML - twierdzi że UTF-8
- ustawienia strumienia wyjściowego - brak

```
1 //response.setCharacterEncoding("UTF-8");
2 response.getWriter()
3     .append("<html>")
4     .append("<head><meta charset=\"UTF-8\">")
5     // .append("<head>")
6     .append("<title>ATJ_homework</title>")
7     .append("</head>")
8     .append("<body>")
9     .append("\nACELN0SZZ\nacelnoszz\n")
10    .append("</body>")
11    .append("</html>");
```

?????  ??? ?????  ???



Servlet Response in UTF-8 (3/4)

- plik java - w UTF-8
- nagłówek HTML - twierdzi że brak
- ustawienia strumienia wyjściowego - UTF-8

```
1 response.setCharacterEncoding("UTF-8");
2 response.getWriter()
3     .append("<html>")
4     //.append("<head><meta charset=\"UTF-8\">")
5     .append("<head>")
6     .append("<title>ATJ_homework</title>")
7     .append("</head>")
8     .append("<body>")
9     .append("\nACELNOSZZ\nacełnoszz\n")
10    .append("</body>")
11    .append("</html>");
```

Ä,,Ä†Ä~Ä^ÄfÄ“ÄšÄ¹Ä» Ä...
Ä†Ä™Ä,Ä,,Ä³Ä>Ä°Ä¼



Servlet Response in UTF-8 (4/4)

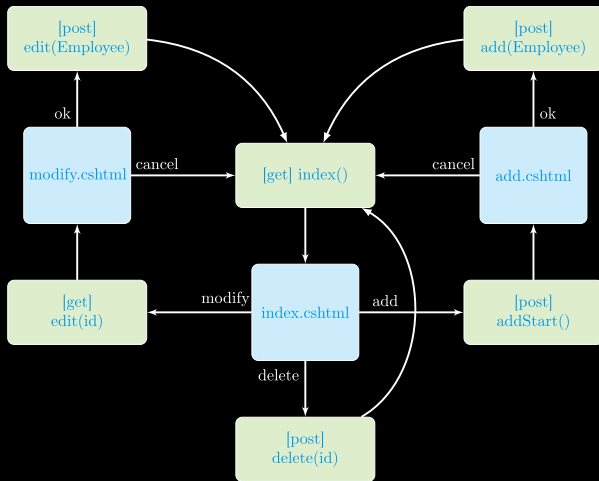
- plik java - w UTF-8
- nagłówek HTML - twierdzi że UTF-8
- ustawienia strumienia wyjściowego - UTF-8

```
1 response.setCharacterEncoding("UTF-8");
2 response.getWriter()
3     .append("<html>")
4     .append("<head><meta charset=\"UTF-8\">")
5     // .append("<head>")
6     .append("<title>ATJ_homework</title>")
7     .append("</head>")
8     .append("<body>")
9     .append("\nACEŁNÓŚZZ\nacełńóśzz\n")
10    .append("</body>")
11    .append("</html>");
```

ĄĆĘŁŃÓŚZZ ąćęłńóśzz



Views (blue) and actions (green) in CRUD





razor files (*.cshtml)

```
1  @model IEnumerable<Company.Models.Employee>
2
3  @{
4      Layout = "_Layout";
5  }
6  <h1>Employee List</h1>
7
8  <div style="text-align:right;margin-right:20px;">
9      @using (Html.BeginForm("AddStart", "Employee", FormMethod.Post))
10     {
11         <input type="submit" class="btn btn-outline-primary" value="Create New" />
12     }
13 </div>
14
15 <div class="table-responsive">
16     <table class="table">
17         <thead>
18             <tr>
19                 <th>
20                     @Html.DisplayNameFor(model => model.FirstName)
21                 </th>
22                 <th>
23                     @Html.DisplayNameFor(model => model.LastName)
24                 </th>
25                 <th>
26                     @Html.DisplayNameFor(model => model.Salary)
27                 </th>
28                 <th>Actions</th>
29             </tr>
30         </thead>
31         <tbody>
32             @foreach (var item in Model)
33             {
34                 <tr>
```




razor files (*.cshtml)

```
35         <td>
36             @Html.DisplayFor(modelItem => item.FirstName)
37         </td>
38         <td>
39             @Html.DisplayFor(modelItem => item.LastName)
40         </td>
41         <td>
42             @Html.DisplayFor(modelItem => item.Salary)
43         </td>
44         <td>
45             <a asp-action="Edit" class="btn btn-outline-primary"
46               asp-route-employeeId ="@item.EmployeeId">Edit</a>
47             <a asp-action="Delete" class="btn btn-outline-danger"
48               asp-route-employeeId ="@item.EmployeeId">Delete</a>
49         </td>
50     </tr>
51 }
52 </tbody>
53 </table>
54 </div>
```

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Views (blue) and
actions (green) in
CRUD

razor files
(*.cshtml)

crud-render

GUI

code

Summary

State



razor files (*.cshtml)

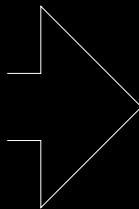
```
<table>
<tr> <th>Imie</th></tr>
<%lista.forEach(e=> { %>
    <tr>
    <td><%= e.firstName %></td>
    </tr>
<%}) ; %>
</table>
```



crud-render

```
html += '<table>'  
html += '<tr> <th>Imie</th></tr>'  
lista.forEach(e=> {  
    html += '<tr>'  
    html += '<td>'+e.firstName+'</td>'  
    html += '</tr>'  
});  
html += '</table>'  
=====
```

```
const persons = [  
    {  
        firstName="Jeremy",  
        lastName="Clarkson",  
    }, {  
        firstName="Richard",  
        lastName="Hammond"  
    }  
];
```

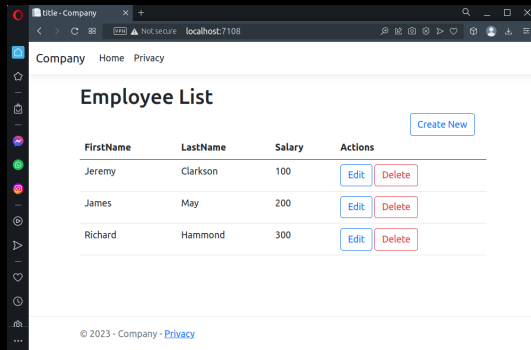
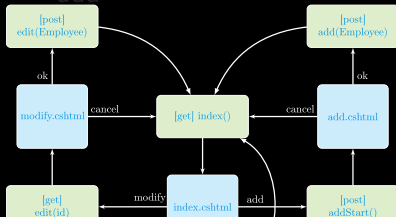


```
<table>  
  <tr> <th>Imie</th></tr>  
  <tr>  
    <td>Jeremy</td>  
  </tr>  
  <tr>  
    <td>Richard</td>  
  </tr>  
</table>
```



GUI

- there is a list on main window
- by pressing a create new button we show view to enter new employee
 - basic verification code prevents sending data to server
 - after providing a values we can do
- now we can pres edit
- the same verification as for add





GUI

- there is a list on main window
- by pressing a create new button we show view to enter new employee
 - basic verification code prevents sending data to server
 - after providing a values we can do
- now we can pres edit
- the same verification as for add

title - Company

Company Home Privacy

Create Employee

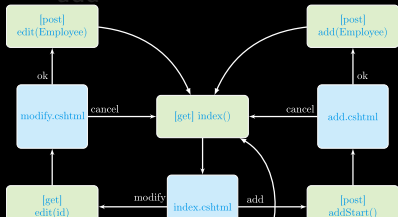
FirstName

LastName

Salary

Save Back

© 2023 - Company - Privacy





GUI

- there is a list on main window
- by pressing a **create new** button we show view to enter new employee
 - **basic verification code prevents sending data to server**
 - after providing a values we can do
- now we can pres edit
- the same verification as for add

title - Company

Company Home Privacy

Create Employee

FirstName

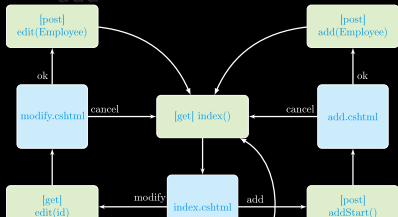
The FirstName field is required.

LastName

The LastName field is required.

Salary

Save Back



```
1 namespace Company.Models;
2 public class Employee
3 {
4     public int? EmployeeId { get; set; }
5     public string FirstName { get; set; }
6     public string LastName { get; set; }
7     public int? Salary { get; set; }
8 }
```



GUI

- there is a list on main window
- by pressing a **create new** button we show view to enter new employee
 - basic verification code prevents sending data to server
 - **after providing a values we can do**
- now we can pres edit
- the same verification as for add

title - Company

Company Home Privacy

Create Employee

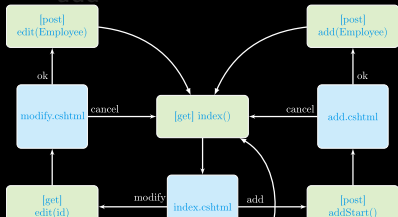
FirstName
Rowan

LastName
Atkinson

Salary
1234

Save Back

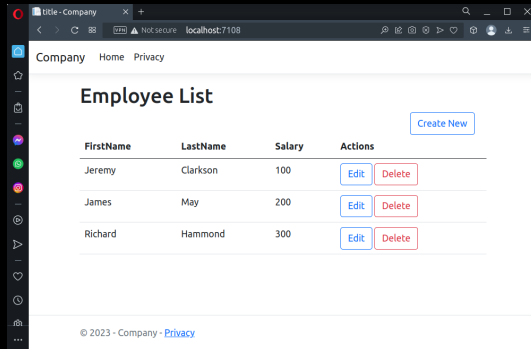
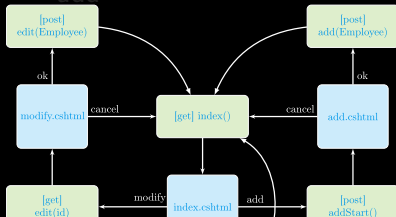
© 2023 - Company - Privacy





GUI

- there is a list on main window
- by pressing a **create new** button we show view to enter new employee
 - basic verification code prevents sending data to server
 - after providing a values we can do
- **now we can pres edit**
- the same verification as for add





GUI

- there is a list on main window
- by pressing a **create new** button we show view to enter new employee
 - basic verification code prevents sending data to server
 - after providing a values we can do
- now we can pres edit
- **the same verification as for add**

title - Company

Company Home Privacy

Edit Employee

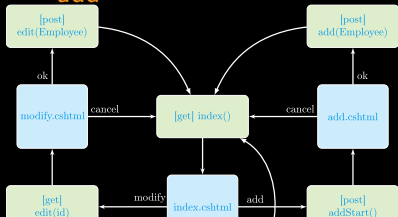
FirstName
Rowan

LastName
Atkinson

Salary

Save Back

© 2023 - Company - Privacy





code

- **Program.cs**

- Models/Employee.cs

- Models/Database.cs

-

Controllers/EmployeeController

- index

- add

- edit

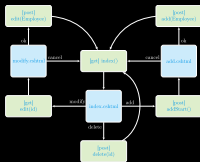
- add

-

View/Employee/index.cshtml

- index

- delete



```

1 using Company.Models;
2 var builder = WebApplication.CreateBuilder(args);
3
4 // Add services to the container.
5 builder.Services.AddSingleton<Database>();
6 builder.Services.AddControllersWithViews();
7 //builder.Services.AddSession();
8
9 var app = builder.Build();
10
11 // Configure the HTTP request pipeline.
12 if (!app.Environment.IsDevelopment())
13 {
14     app.UseExceptionHandler("/Home/Error");
15     // The default HSTS value is 30 days.
16     app.UseHsts();
17 }
18
19 app.UseHttpsRedirection();
20 app.UseStaticFiles();
21 //app.UseSession();
22 app.UseRouting();
23
24 app.UseAuthorization();
25
26 app.MapControllerRoute(
27     name: "default",
28     pattern: "{controller=Employee}/{action=Index}/{id?}");
29
30 app.Run();
  
```



code

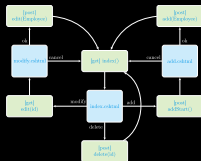
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- edit
- add

View/Employee/index.cshtml

- index
- delete



```

1 namespace Company.Models;
2
3 public class Employee
4 {
5     public Employee()
6     {
7         EmployeeId = default;
8         FirstName = "";
9         LastName = "";
10        Salary = default;
11    }
12
13    public int? EmployeeId { get; set; }
14    public string FirstName { get; set; }
15    public string LastName { get; set; }
16    public int? Salary { get; set; }
17 }

```



code

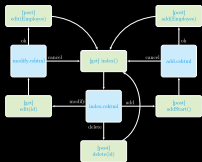
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- edit
- add

View/Employee/index.cshtml

- index
- delete



```

1 namespace Company.Models;
2
3 public interface IDatabase
4 {
5     List<Employee> Employees { get; set; }
6 }
7
8 public class Database:IDatabase
9 {
10     public Database()
11     {
12         Employees = new List<Employee>()
13     {
14         new Employee { EmployeeId = 0, FirstName = "Jeremy",
15             ↳ LastName = "Clarkson", Salary = 100 },
16         new Employee { EmployeeId = 1, FirstName = "James",
17             ↳ LastName = "May", Salary = 200 },
18         new Employee { EmployeeId = 2, FirstName = "Richard",
19             ↳ LastName = "Hammond", Salary = 300 }
20     };
21 }
22
23 public List<Employee> Employees { get; set; }
24
25 public bool Add(Employee e)
26 {
27     Employees.Add(new Employee { EmployeeId = GenerateId(),
28         ↳ FirstName = e.FirstName, LastName = e.LastName,
29         ↳ Salary = e.Salary });
30     return true;
31 }
32
33 public bool Modify(Employee e)
34 {
35     Employee? ob = Employees.FirstOrDefault(i =>
36         i.EmployeeId == e.EmployeeId);
37 }
  
```



code

- Program.cs
- Models/Employee.cs
- Models/Database.cs

•

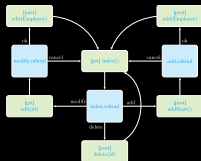
Controllers/EmployeeController

- index
- add
- edit
- add

•

View/Employee/index.cshtml

- index
- delete



```

35     ob.Salary = e.Salary;
36     return true;
37 }
38
39 public bool Delete(Employee e)
40 {
41     Employee? ob = Employees.FirstOrDefault(i =>
42         i.EmployeeId == e.EmployeeId);
43     if (ob == null)
44         return false;
45     Employees.Remove(ob);
46     return true;
47 }
48
49 public int GenerateId()
50 {
51     return Employees.Max(r =>
52         r.EmployeeId).GetValueOrDefault(0)+1;
53 }
  
```



code

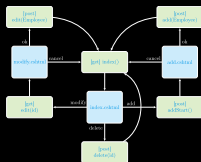
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- edit
- add

View/Employee/index.cshtml

- index
- delete



```

1  using Company.Models;
2  using Microsoft.AspNetCore.Mvc;
3  namespace Company.Controllers;
4
5  public class EmployeeController : Controller
6  {
7      private readonly Database _database;
8
9      public EmployeeController(Database db)
10     {
11         _database = db;
12     }
13
14     public IActionResult Index()
15     {
16         return View(_database.Employees);
17     }
18     ...
19 }
  
```

code

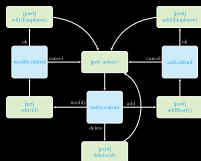
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- edit
- add

View/Employee/index.cshtml

- index
- delete



```

1  @model IEnumerable<Company.Models.Employee>
2
3  @{
4      Layout = "_Layout";
5  }
6  <h1>Employee List</h1>
7
8  <div style="text-align:right;margin-right:20px;">
9      @using (Html.BeginForm("AddStart", "Employee",
10         ↪ FormMethod.Post)) {
11         <input type="submit" class="btn btn-outline-primary"
12         ↪ value="Create New"/>
13     }
14 </div>
15
16 <div class="table-responsive">
17     <table class="table">
18         <thead>
19             <tr>
20                 <th>
21                     @Html.DisplayNameFor(model =>
22                     ↪ model.FirstName)
23                 </th>
24                 <th>
25                     @Html.DisplayNameFor(model =>
26                     ↪ model.LastName)
27                 </th>
28                 <th>
29                     @Html.DisplayNameFor(model => model.Salary)
30                 </th>
31                 <th>Actions</th>
32             </tr>
33         </thead>
34         <tbody>
35             @foreach (var item in Model)
36             {

```



code

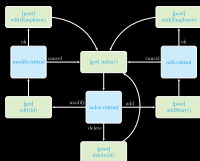
- Program.cs 35
- Models/Employee.cs 36
- Models/Database.cs 37
- 38

Controllers/EmployeeController

- index 41
- add 42
- edit 43
- add 44
- 45

View/Employee/index.cshtml

- index 48
- delete 49



```

@Html.DisplayFor(modelItem =>
    item.FirstName)
</td>
<td>
    @Html.DisplayFor(modelItem =>
        item.LastName)
</td>
<td>
    @Html.DisplayFor(modelItem =>
        item.Salary)
</td>
<td>
    <a asp-action="Edit" class="btn
    btn-outline-primary"
    asp-route-employeeId
    => "@item.EmployeeId">Edit</a>
    <a asp-action="Delete" class="btn
    btn-outline-danger"
    asp-route-employeeId
    => "@item.EmployeeId">Delete</a>
</td>
</tr>
}
</tbody>
</table>
</div>
  
```




code

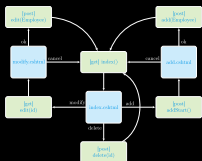
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- edit
- add

View/Employee/index.cshtml

- index
- delete



```

1 [HttpPost]
2 public IActionResult AddStart()
3 {
4     ViewBag.PageName = "Create Employee" ;
5     ViewBag.ActionName = "Add" ;
6     ViewBag.IsEdit = false ;
7     return View("AddOrEdit", new Employee{EmployeeId =
8         ↳ _database.GenerateId()});
9 }
10
11 [HttpPost]
12 [ValidateAntiForgeryToken]
13 public IActionResult Add(
14     ↳ [Bind("EmployeeId,FirstName,LastName,Salary")]
15     ↳ Employee employeeData)
16 {
17     if (ModelState.IsValid)
18     {
19         _database.Add(employeeData);
20         return RedirectToAction(nameof(Index));
21     }
22     ViewBag.PageName = "Create Employee" ;
23     ViewBag.ActionName = "Add" ;
24     ViewBag.IsEdit = false ;
25     return View("AddOrEdit", employeeData);
26 }
  
```



code

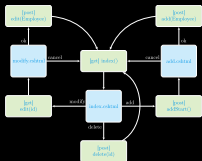
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- edit
- add

View/Employee/index.cshtml

- index
- delete



```

1  @model Company.Models.Employee
2
3  @{
4      Layout = "_Layout";
5  }
6
7  <div class="container p-3 my-3 border" >
8      <h1> @ViewBag.PageName</h1>
9
10     <div class="row">
11         <div class="col-sm-6">
12             <hr />
13             <form asp-action="@ViewBag.ActionName">
14                 <div asp-validation-summary="ModelOnly"
15                     class="text-danger"></div>
16                 @if (@ViewBag.IsEdit)
17                 {
18                     <input type="hidden"
19                         class="form-control"
20                         asp-for="EmployeeId"/>
21                 }
22                 <div class="form-group">
23                     <label asp-for="FirstName"></label>
24                     <input asp-for="FirstName"
25                         class="form-control">
26                     <span asp-validation-for="FirstName"
27                         class="text-danger"></span>
28                 </div>
29                 <div class="form-group">
30                     <label asp-for="LastName"></label>
31                     <input asp-for="LastName"
32                         class="form-control">
33                     <span asp-validation-for="LastName"
34                         class="text-danger"></span>
35                 </div>
36             </form>
37         </div>
38     </div>
39 </div>

```



code

- Program.cs
- Models/Employee.cs
- Models/Database.cs

•

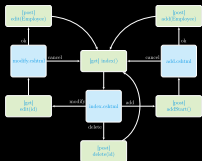
Controllers/EmployeeController

- index
- add
- edit
- add

•

View/Employee/index.cshtml

- index
- delete



35

36

37

38

39

40

41

42

43

44

45

```
<div class="float-end">
```

```
<button style="margin-top:10px"
```

```
< class="btn btn-primary"
```

```
< type="submit">Save</button>
```

```
<a style="margin-top:10px" class="btn
```

```
< btn-danger"
```

```
< asp-action="Index">Back</a>
```

```
</div>
```

```
</form>
```

```
</div>
```

```
</div>
```

```
@section Scripts {
```

```
@{await
```

```
< Html.RenderPartialAsync("_ValidationScriptsPartial");}
```

```
<
```

```
<
```



code

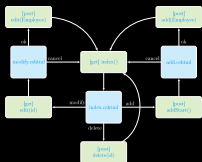
- Program.cs
- Models/Employee.cs
- Models/Database.cs

Controllers/EmployeeController

- index
- add
- **edit**
- ~~add~~

View/Employee/index.cshtml

- index
- delete



```

1 public async Task<IActionResult> Edit(int? employeeId)
2 {
3     ViewBag.PageName = "Edit Employee" ;
4     ViewBag.ActionName = "Edit" ;
5     ViewBag.IsEdit = true ;
6     Employee employee = await
7         ↳ Task.Run(() => _database.Employees.Find(i =>
8             ↳ i.EmployeeId == employeeId));
9     return View("AddOrEdit", employee);
10 }
11 [HttpPost]
12 [ValidateAntiForgeryToken]
13 public async Task<IActionResult>
14     ↳ Edit([Bind("EmployeeId,FirstName,LastName,Salary")]
15         Employee employeeData)
16 {
17     if (ModelState.IsValid)
18     {
19         Employee employee = await Task.Run(() =>
20             ↳ _database.Employees.Find(i => i.EmployeeId ==
21                 ↳ employeeData.EmployeeId));
22         employee.FirstName = employeeData.FirstName;
23         employee.LastName = employeeData.LastName;
24         employee.Salary = employeeData.Salary;
25         return RedirectToAction(nameof(Index));
26     }
27 }
28 ViewBag.PageName = "Edit Employee" ;
29 ViewBag.ActionName = "Edit" ;
30 ViewBag.IsEdit = true ;
31 return View("AddOrEdit", employeeData);

```



code

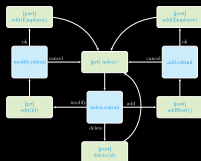
- Program.cs
- Models/Employee.cs
- Models/Database.cs

- Controllers/EmployeeController

- index
- add
- edit
- **add**

- View/Employee/index.cshtml

- index
- delete



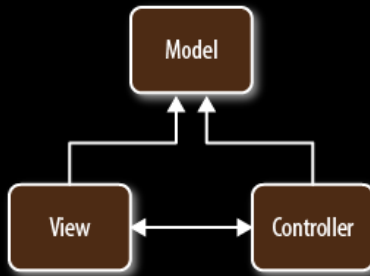
```

1 public async Task<IActionResult> Delete(int employeeId)
2 {
3     Employee employee = await Task.Run(() =>
4         ↪ _database.Employees.Find(i => i.EmployeeId ==
5         ↪ employeeId));
6     _database.Delete(employee);
    return RedirectToAction(nameof(Index));
}

```



MVC



- we forget about automatic state saving
- we could customize routing - so presented path may not be path to the view
- views may be created in different languages ([Razor](#), [ASPX](#))
- we create an object and set their properties from the parameters
- for simple parameters we fit accordingly
- address path is also a parameter source



MVC:How it Works

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

MVC

MVC:How it Works

Code Behind

Structure

many projects

Model

ViewData and

ViewBag

Routing

układy

partial views

postman

State





MVC.NET - a piece of code

model

```
1 namespace Mvc4App.Models {  
2     public class Person {  
3         public string FirstName { get; set; }  
4         public string LastName { get; set; }  
5     }  
6 }
```

controller

```
1 namespace Mvc4App.Controllers {  
2     public class PersonController : Controller {  
3         public ActionResult Index() {  
4             Person p = new Person();  
5             return View(p);  
6         }  
7         [HttpPost]  
8         public ActionResult Index(Person p) {  
9             return View(p);  
10        }  
11    }  
12 }
```





MVC.NET - view

```
1 @model Mvc4App.Models.Person
2 @{ Layout = null; }
3 <!DOCTYPE html>
4 <html><head><meta name="viewport" content="width=device-width" />
5   <title>Index</title></head>
6   <body>
7     @using (Html.BeginForm()) {
8       @Html.AntiForgeryToken() @Html.ValidationSummary(true)
9       <fieldset>
10        <legend>Person</legend>
11        <div class="editor-label">
12          @Html.LabelFor(model => model.FirstName)
13        </div>
14        <div class="editor-field">
15          @Html.EditorFor(model => model.FirstName)
16          @Html.ValidationMessageFor(model => model.FirstName)
17        </div>
18        <p> <input type="submit" value="Save" /> </p>
19      </fieldset>
20    }
21    <div> @Html.ActionLink("Back to List", "Index") </div>
22  </body>
23 </html>
```



MVC.NET - controller without view

```
1 public ActionResult ViewPerson(Person p) {  
2     return new ActionResult {  
3         ContentType = "text/html",  
4         StatusCode = 200, //(int)HttpStatusCode.OK,  
5         Content = @"<!DOCTYPE html>  
6 <html>  
7 <head><meta charset=""UTF-8""><title>EGUI</title></head>  
8 <body>  
9 <h2>Formy</h2>  
10  
11 <form action=""/Home/Person"" method=""post"">  
12     First Name:<br>  
13     <input type=""text"" name=""FirstName"" value=""Johny""> <br>  
14     Last Name:<br>  
15     <input type=""text"" name=""LastName"" value=""Bean""> <br><br>  
16     <input type=""submit"" name = 'btn' value=""Send"">  
17     <input type=""submit"" name = 'btn' value=""Cancel."">  
18 </form>  
19 </body>  
20 </html>"  
21     };  
22 }
```

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

MVC

MVC:How it Works

Code Behind

Structure

many projects

Model

ViewData and

ViewBag

Routing

układy

partial views

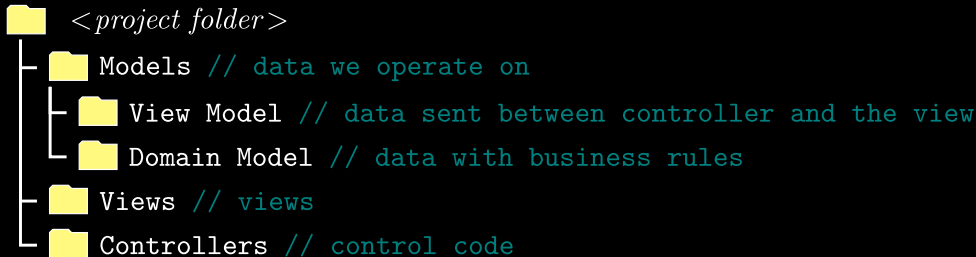
postman

State



MVC Folders

Idea



main goal: separation:

- lower complexity
- better testability because of elements replacement



Old approach: Page Controller Pattern

without architecture

- everything in event handling procedures
- smart controls reduces code, but not enough
- poor testability

second approach - layers

- typically GUI, data access layer, database
- still poor testability
- poor standarization



Advantages of MVC ...

we split

- request handling
- display handling

advantages

- split
- testability
- dynamic controll over page layout
- *asp file may serve as an view, but it do not contain code

We have MVC version 5.0

We create a new project, we have:

- ASP.NET MVC 4 Web Application
- Empty Web Application

First will be a question about testing model



Structure

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

MVC
MVC:How it Works

Code Behind

Structure

many projects

Model

ViewData and
ViewBag

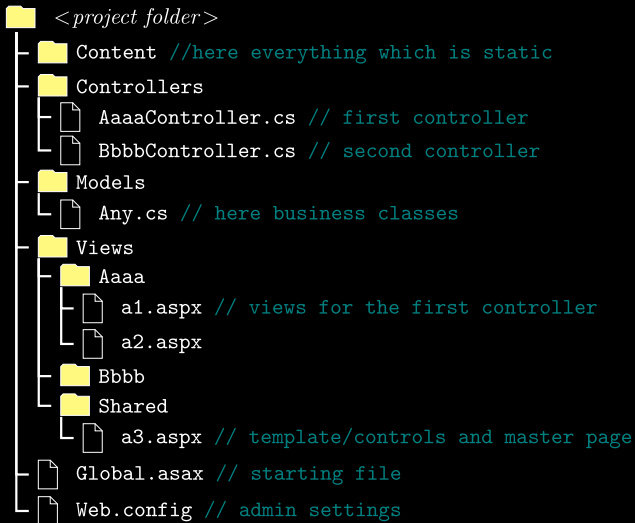
Routing

układy

partial views

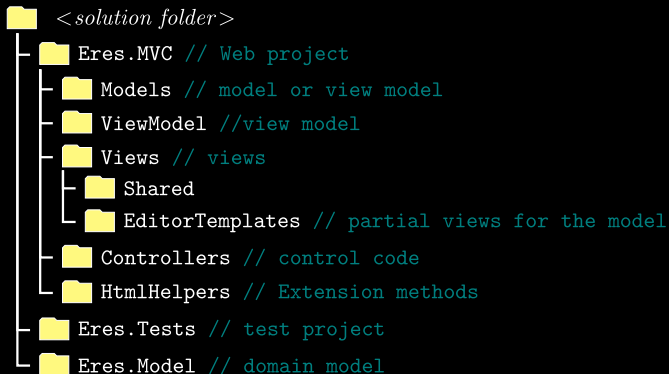
postman

State





MVC-Many projects



- Separate model can be reused
- Automatic testing should cover both model and controller



The Model

- Data can be stored any way
- But why do not use **Entity Framework??**
- Using adnotations we could attach error checking in the model

```
1 public class CalcData {  
2     [Required(ErrorMessage = "Please enter Arg1")]  
3     public string Arg1 { get; set; }  
4     [Required(ErrorMessage = "Please enter Arg0")]  
5     public string Arg0 { get; set; }  
6     [Required(ErrorMessage = "Please enter Op")]  
7     [RegularExpression(".", ErrorMessage = "Please enter Op")]  
8     public string Op { get; set; }  
9     public string DisplayValue { get; set; }  
10 }
```




ViewData and ViewBag

ViewData

- object of type `ViewDataDictionary`
- we can write: `ViewData["CurrentTime"] = DateTime.Now;`

ViewBag

- uses internally `ViewData`
- we can write `ViewBag.CurrentTime = DateTime.Now;`

Ograniczenia

Constraint - `ViewBag.xxx` is a dynamic type (`dynamic`). So extending methods do not know the type

- `@Html.TextBox("name", ViewBag.Name)` not working
- `@Html.TextBox("name", ViewData["Name"])` working
- `@Html.TextBox("name", (string)ViewBag.Name)` working



Routing

Global.asax

```
1 public class MvcApplication : System.Web.HttpApplication {
2     public static void RegisterRoutes(RouteCollection routes) {
3         routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
4         routes.MapRoute(
5             "Default", // Route name
6             "{controller}/{action}/{id}", // URL with parameters
7             new { controller = "Home",
8                 action = "Index",
9                 id = UrlParameter.Optional } // Parameter defaults
10        );
11    }
12    protected void Application_Start() {
13        AreaRegistration.RegisterAllAreas();
14        RegisterRoutes(RouteTable.Routes);
15    }
16 }
```

- <http://localhost:49204/>
- <http://localhost:49204/Home>
- <http://localhost:49204/Home/Index>
- <http://localhost:49204/Home/Info>





Layouts - sample layout

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="utf-8" />
5   <title>@View.Title</title>
6 </head>
7 <body>
8   <div class="header">
9     @RenderSection("Header")
10  </div>
11  @RenderBody()
12  @if(IsSectionDefined("Footer")) {
13    <div class="footer">
14      @RenderSection("Footer", required: false)
15    </div>
16  }
17  <ul class="nav navbar-nav">
18    <li>@Html.ActionLink("Home", "Index", "Home")</li>
19  </ul>
20  @Scripts.Render("~/bundles/jquery")
21  @Scripts.Render("~/bundles/bootstrap")
22  @RenderSection("scripts", required: false)
23 </body>
24 </html>
```





Layouts - page referring to the layout

```
1 @{ Layout = "~//_Layout.cshtml"; }
2 @section Header {
3 <h1>EGUI</h1>
4 }
5 @section Footer {
6 Copyright 2016
7 }
8 <div class="main">
9 here should be a program ...
10 </div>
```





partial views

```
1 @{
2     Layout = "~//_Layout.cshtml";
3     var data = MyApp.GetData();
4 }
5 @section Header {
6     <h1>NTR</h1>
7 }
8 @foreach(var item in data) {
9     var url = "http://localhost:1111/MyApp/data.cshtml?id=" + item.ID;
10    <div>
11        <h3>@item.Name</h3>
12        <div>@item.Description</div>
13        @TwitterHelpers.TweetButton( url: url, text: @post.Title )
14        @Facebook.LikeButton( href: url )
15    </div>
16 }
```





partial views - a code after applying the view

```
1  @{
2      Layout = "~//_Layout.cshtml";
3      var data = MyApp.getData();
4  }
5  @section Header {
6      <h1>NTR</h1>
7  }
8  @foreach(var item in data) {
9      RenderPage("Data/_Data.cshtml", new { Data = item })
10     //RenderPage("Data/_Data.cshtml", item) (2)
11 }
```





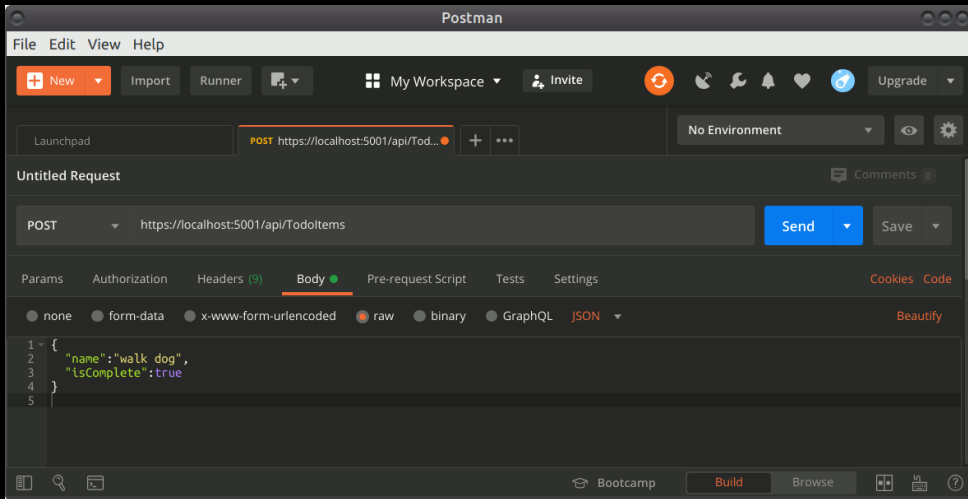
partial views - code of the view

```
1 @{
2     var item = Data.Post;
3     //var item = PageData[1]; (2)
4     var url = "http://localhost:1111/MyApp/data.cshtml?id=" + item.ID;
5 }
6 <div>
7     <h3>@item.Name</h3>
8     <div>@item.Description</div>
9         @TwitterHelpers.TweetButton( url: url, text: @post.Title )
10         @Facebook.LikeButton( href: url )
11     </div>
12 </div>
```



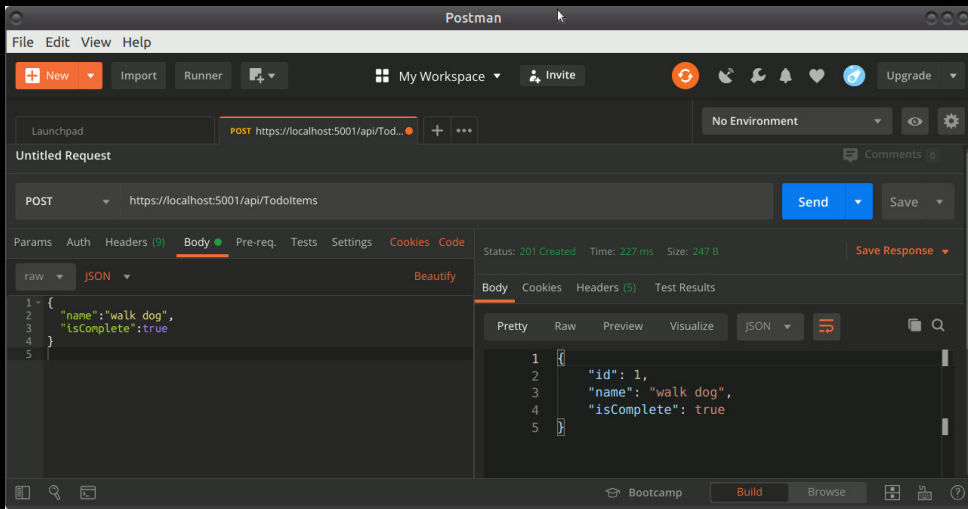


postman - sending HTTP requests





postman - sending HTTP requests





postman - sending HTTP requests

The screenshot shows the Postman application window. The top bar includes the menu (File, Edit, View, Help) and a toolbar with buttons for New, Import, Runner, and My Workspace. The main area displays a POST request to `https://localhost:5001/api/TodoItems`. The request body is set to JSON and contains the following content:

```
1 {
2   "name": "walk dog",
3   "isComplete": true
4 }
5
```

The right sidebar shows the response details, including the status (201 Created), time (227 ms), and size (247 B). The Headers tab is selected, showing the following headers:

KEY	VALUE
Date	Thu, 16 Apr 2020 17:55:38 GMT
Content-Type	application/json; charset=utf-8
Server	Kestrel
Transfer-Encoding	chunked
Location	https://localhost:5001/api/TodoItems/1



State Management (1)

- HTTP is a stateless protocol.
- By default, HTTP requests are independent messages that don't retain user values

storage	storage mechanism
Cookies	HTTP cookies. May include data stored using server-side app code.
Session state	HTTP cookies and server-side app code
TempData	HTTP cookies or session state
Query strings	HTTP query strings
Hidden fields	HTTP form fields
HttpContext.Items	Server-side app code
Cache	Server-side app code



State Management (2) session configuration

`Microsoft.AspNetCore.Session` package:

- To enable the session middleware, Startup must contain:



State Management (3) session configuration

Microsoft.AspNetCore.Session package:

- To enable the session middleware, Startup must contain:
 - Any of the IDistributedCache memory caches. The IDistributedCache implementation is used as a backing store for session.

```
1 public void ConfigureServices(IServiceCollection services) {  
2  
3  
4  
5  
6  
7  
8     services.AddControllersWithViews();  
9 }
```



State Management (4) session configuration

Microsoft.AspNetCore.Session package:

- To enable the session middleware, Startup must contain:
 - Any of the IDistributedCache memory caches. The IDistributedCache implementation is used as a backing store for session.

```
1 public void ConfigureServices(IServiceCollection services) {  
2     services.AddDistributedMemoryCache();  
3  
4  
5  
6  
7  
8     services.AddControllersWithViews();  
9 }
```



State Management (5) session configuration

Microsoft.AspNetCore.Session package:

- To enable the session middleware, Startup must contain:
 - Any of the IDistributedCache memory caches. The IDistributedCache implementation is used as a backing store for session.
 - A call to AddSession in ConfigureServices.

```
1 public void ConfigureServices(IServiceCollection services) {  
2     services.AddDistributedMemoryCache();  
3  
4  
5  
6  
7  
8     services.AddControllersWithViews();  
9 }
```



State Management (6) session configuration

Microsoft.AspNetCore.Session package:

- To enable the session middleware, Startup must contain:
 - Any of the IDistributedCache memory caches. The IDistributedCache implementation is used as a backing store for session.
 - A call to AddSession in ConfigureServices.

```
1 public void ConfigureServices(IServiceCollection services) {  
2     services.AddDistributedMemoryCache();  
3     services.AddSession(options => {  
4         options.IdleTimeout = TimeSpan.FromSeconds(10);  
5         options.Cookie.HttpOnly = true;  
6         options.Cookie.IsEssential = true;  
7     });  
8     services.AddControllersWithViews();  
9 }
```




State Management (7) session configuration

Microsoft.AspNetCore.Session package:

- To enable the session middleware, Startup must contain:
 - Any of the IDistributedCache memory caches. The IDistributedCache implementation is used as a backing store for session.
 - A call to AddSession in ConfigureServices.
 - A call to UseSession in Configure.

```
1 public void Configure(IApplicationBuilder app, IWebHostEnvironment env) {  
2     ...  
3     app.UseAuthorization();  
4  
5     app.UseEndpoints(endpoints => {  
6         endpoints.MapControllerRoute(  
7             name: "default",  
8             pattern: "{controller=Home}/{action=Index}/{id?}");  
9     });  
10 }
```



State Management (8) session configuration

Microsoft.AspNetCore.Session package:

- To enable the session middleware, Startup must contain:
 - Any of the IDistributedCache memory caches. The IDistributedCache implementation is used as a backing store for session.
 - A call to AddSession in ConfigureServices.
 - A call to UseSession in Configure.

```
1 public void Configure(IApplicationBuilder app, IWebHostEnvironment env) {  
2     ...  
3     app.UseAuthorization();  
4     app.UseSession();  
5     app.UseEndpoints(endpoints => {  
6         endpoints.MapControllerRoute(  
7             name: "default",  
8             pattern: "{controller=Home}/{action=Index}/{id?}");  
9     });  
10 }
```



Session Example

- we serialize the object to the string ourselves

```
1 using System.Text.Json;           // serialisation to string
2 ...
3 private const string MEMORY = "Memory";
4 Stack<Calc> memory {
5     get {
6         return JsonSerializer.Deserialize<Stack<Calc> >(HttpContext.Session.GetString(MEMORY));}
7     set {HttpContext.Session.SetString(MEMORY, JsonSerializer.Serialize(value));}
8 }
9
10 [HttpPost]
11 public IActionResult Index(Calc calc) {
12     var memory = this.memory;
13     memory.Push(calc);
14     this.memory = memory;
15     return View(calc);
16 }
```



Session Example

- we serialize the object to the string ourselves
- you have to handle the case when the session expires

```
1 using System.Text.Json;           // serialisation to string
2 ...
3 private const string MEMORY = "Memory";
4 Stack<Calc> memory {
5     get {
6         return JsonSerializer.Deserialize<Stack<Calc> >(HttpContext.Session.GetString(MEMORY));}
7     set {HttpContext.Session.SetString(MEMORY, JsonSerializer.Serialize(value));}
8 }
9
10 [HttpPost]
11 public IActionResult Index(Calc calc) {
12     var memory = this.memory;
13     memory.Push(calc);
14     this.memory = memory;
15     return View(calc);
16 }
```



Session Extension

- we can create extension methods for all types:

```
1 public static class SessionExtensions {
2     public static void Set<T>(this ISession session, string key, T value) {
3         session.SetString(key, JsonSerializer.Serialize(value));
4     }
5     public static T Get<T>(this ISession session, string key) {
6         var value = session.GetString(key);
7         return value == null ? default : JsonSerializer.Deserialize<T>(value);
8     }
9 }
```



Session Extension

- we can create extension methods for all types:
- using we do not see serialisation

```
1 public static class SessionExtensions {  
2     public static void Set<T>(this ISession session, string key, T value) {  
3         session.SetString(key, JsonSerializer.Serialize(value));  
4     }  
5     public static T Get<T>(this ISession session, string key) {  
6         var value = session.GetString(key);  
7         return value == null ? default : JsonSerializer.Deserialize<T>(value);  
8     }  
9 }
```

example:

```
1 // set time if has not been set yet  
2 if (HttpContext.Session.Get<DateTime>(SessionKeyTime) == default) {  
3     HttpContext.Session.Set<DateTime>(SessionKeyTime, currentTime);  
4 }
```



TempData Example

- TempData stores the data temporarily and automatically removes it after retrieving a value.

```
1 private const string MEMORY = "Memory";
2 Stack<Calc> memory {
3     get { return TempData.Get<Stack<Calc> >(MEMORY);} // Get is mine!
4     set { TempData.Put<Stack<Calc> >(MEMORY, value);} // Put is mine!
5 }
6
7 [HttpPost]
8 public IActionResult Index(Calc calc) {
9     var memory = this.memory;
10    memory.Push(calc);
11    this.memory = memory;
12    return View(calc);
13 }
```



TempData Example

- TempData stores the data temporarily and automatically removes it after retrieving a value.
- to read key "Memory" in *.cshtml we can write: `@TempData["Memory"]`

```
1 private const string MEMORY = "Memory";
2 Stack<Calc> memory {
3     get { return TempData.Get<Stack<Calc> >(MEMORY);} // Get is mine!
4     set { TempData.Put<Stack<Calc> >(MEMORY, value);} // Put is mine!
5 }
6
7 [HttpPost]
8 public IActionResult Index(Calc calc) {
9     var memory = this.memory;
10    memory.Push(calc);
11    this.memory = memory;
12    return View(calc);
13 }
```




TempData Example

- TempData stores the data temporarily and automatically removes it after retrieving a value.
- to read key "Memory" in *.cshtml we can write: `@TempData["Memory"]`
- for complex types we need serialisation and extension functions

```
1 using System.Text.Json;
2 using Microsoft.AspNetCore.Mvc.ViewFeatures;
3 namespace code.Extensions {
4     public static class TempDataHelper {
5         public static void Put<T>(this IDictionary tempData,
6                                 string key, T value) where T : class {
7             tempData[key] = JsonSerializer.Serialize(value);
8         }
9         public static T Get<T>(this IDictionary tempData,
10                               string key) where T : class {
11             tempData.TryGetValue(key, out object o);
12             return o == null ? null : JsonSerializer.Deserialize<T>((string)o);
13         }
14         public static T Peek<T>(this IDictionary tempData,
15                                string key) where T : class {
16             object o = tempData.Peek(key);
17             return o == null ? null : JsonSerializer.Deserialize<T>((string)o);
18         }
19     }
20 }
```



TempData Provider

- by default we use cookie TempData provider

```
1 // This ... Use this method to add services to the container.
2 public void ConfigureServices(IServiceCollection services) {
3     services.AddDistributedMemoryCache();
4
5
6
7
8
9     services.AddControllersWithViews();
10
11 }
```



TempData Provider

- by default we use cookie TempData provider
- we can use session state TempData provider

```
1 // This ... Use this method to add services to the container.
2 public void ConfigureServices(IServiceCollection services) {
3     services.AddDistributedMemoryCache();
4     services.AddSession(options => {
5         options.IdleTimeout = TimeSpan.FromSeconds(1000);
6         options.Cookie.HttpOnly = true;
7         options.Cookie.IsEssential = true;
8     });
9     services.AddControllersWithViews()
10         .AddSessionStateTempDataProvider();
11 }
```



Sample application which saves history

[calc](#) [Home](#) [Privacy](#)

arg[0]

0

arg[1]

0

Select Operation:

Wynik:

0

Historia:0

Back

Policz

© 2020 - calc - Privacy



Startup.cs

```
1 using System;
2 using Microsoft.AspNetCore.Builder;
3 using Microsoft.AspNetCore.Hosting;
4 using Microsoft.Extensions.Configuration;
5 using Microsoft.Extensions.DependencyInjection;
6 using Microsoft.Extensions.Hosting;
7
8 namespace calc {
9     public class Startup {
10         public Startup(IConfiguration configuration) {
11             Configuration = configuration;
12         }
13
14         public IConfiguration Configuration { get; }
15
16         // This ... Use this method to add services to the container.
17         public void ConfigureServices(IServiceCollection services) {
18             services.AddDistributedMemoryCache();
19             services.AddSession(options => {
20                 options.IdleTimeout = TimeSpan.FromSeconds(1000);
21                 options.Cookie.HttpOnly = true;
22                 options.Cookie.IsEssential = true;
23             });
24             services.AddControllersWithViews();
25 }
```



Startup.cs



Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

State

state
session state
session example
extension
TempData
TempData
Provider
rzzut ekranu

Startup.cs

Calc.cs
HomeController.cs
index.cshtml
Area folder
structure
Scaffold

```
26 // This ... Use this method to configure the HTTP request pipeline.
27 public void Configure(IApplicationBuilder app, IWebHostEnvironment env) {
28     if (env.IsDevelopment()) {
29         app.UseDeveloperExceptionPage();
30     } else {
31         app.UseExceptionHandler("/Home/Error");
32         // The default HSTS ... see https://aka.ms/aspnetcore-hsts.
33         app.UseHsts();
34     }
```



Startup.cs



Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

State

state
session state
session example
extension
TempData
TempData
Provider
rzzut ekranu

Startup.cs

Calc.cs

HomeController.cs

index.cshtml

Area folder

structure

Scaffold

```
35     app.UseHttpsRedirection();
36     app.UseStaticFiles();
37
38     app.UseRouting();
39
40     app.UseAuthorization();
41     app.UseSession();
42     app.UseEndpoints(endpoints => {
43         endpoints.MapControllerRoute(
44             name: "default",
45             pattern: "{controller=Home}/{action=Index}/{id?}");
46     });
47 }
48 }
49 }
```



Models/Calc.cs

```
1 using System.Text.Json.Serialization;
2
3 namespace calc.Models {
4     public enum Operation {
5         Plus,
6         Minus
7     }
8
9     public class Calc {
10         public Calc() {
11             arg = new int[2];
12         }
13         public int[] arg { get; set; }
14         public Operation op { get; set; }
15
16         [JsonIgnore] // property will not be serialized
17         public int value { get { return op==Operation.Plus?arg[0]+arg[1]:arg[0]-arg[1];}}
18     }
19 }
```




Controllers/HomeController.cs

```
1 using System.Collections.Generic;
2 using System.Diagnostics;
3 using Microsoft.AspNetCore.Http;
4 using Microsoft.AspNetCore.Mvc;
5 using Microsoft.Extensions.Logging;
6 using System.Text.Json;
7 using calc.Models;
8
9 namespace calc.Controllers {
10     public class HomeController : Controller {
11         private readonly ILogger<HomeController> _logger;
12         private const string MEMORY = "Memory";
13
14         Stack<Calc> memory {
15             get {
16                 return JsonSerializer.Deserialize<Stack<Calc>>(HttpContext.Session.GetString(MEMORY));
17             }
18             set {HttpContext.Session.SetString(MEMORY, JsonSerializer.Serialize(value));}
19         }
20         public HomeController(ILogger<HomeController> logger) {
21             _logger = logger;
22         }
23         public IActionResult Index() {
24             var memory = this.memory = new Stack<Calc>();
25             ViewBag.Count = memory.Count; // przekazanie danych przez ViewBag
26             return View(new Calc());
27         }
28     }
29 }
```



Controllers/HomeController.cs

```
27 [HttpPost]
28 public IActionResult Index(Calc calc) {
29     var memory = this.memory;
30     if(ModelState.IsValid)
```



Controllers/HomeController.cs

```
31         memory.Push(calc);
32         this.memory = memory;
33         ViewBag.Count = memory.Count;           // data sent using ViewBag
34         return View(calc);
35     }
36     public IActionResult Back(Calc calc) {
37         var memory = this.memory;
38         if(memory.Count > 0)
39             calc = memory.Pop();
40         this.memory = memory;                     // TempData require writing
41         ViewBag.Count = memory.Count;             // data sent using ViewBag
42         return View("Index", calc);               // going to "Index" instead of "Back"
43     }
44     [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]
45     public IActionResult Error() {
46         return View(new ErrorViewModel {
47             RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier
48         });
49     }
50 }
51 }
```

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

State

state
session state
session example

extension

TempData

TempData

Provider

rzut ekranu

Startup.cs

Calc.cs

HomeController.cs

index.cshtml

Area folder

structure

Scaffold



Views/Home/index.cshtml

```
1  @using calc.Models
2  @model Calc
3  @{
4      ViewData["Title"] = "Calculator";
5  }
6
7  @using(Html.BeginForm("index","Home")) { // we always return to index action
8      @Html.AntiForgeryToken()
9      @Html.ValidationSummary(true, "", new { @class = "text-danger" } )
10
11     @for(int i=0;i<2;i++) {
12         <div class="form-group">
13             @Html.LabelFor(balbinka=>balbinka.arg[i])
14             @Html.TextBoxFor(model=>model.arg[i], new { @class="form-control" })
15             @Html.ValidationMessageFor(Model=>Model.arg[i],"", new { @class = "text-danger" })
16         </div>
17     }
18
19     <div class="form-group">
20         @Html.Label("Select Operation:", "Select Operation:",
21             new { @class = "col-md-2 control-label" })
22         <div class="col-md-4">
23             @Html.DropDownList("Name", new SelectList(Enum.GetValues(typeof(Operation))), "",
24                 new { @class = "form-control" })
25         </div>
26     </div>
```



Views/Home/index.cshtml

```
27  
28 <div class="form-group">  
29     @Html.Label("Wynik:", "Wynik:", new { @class = "control-label" })  
30     @Html.TextBoxFor(model => model.value, new { @class = "form-control", @readonly = true })
```

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

State

state
session state
session example

extension

TempData

TempData

Provider

rzut ekranu

Startup.cs

Calc.cs

HomeController.cs

index.cshtml

Area folder

structure

Scaffold



Views/Home/index.cshtml

```
31     </div>
32
33     <div class="form-group">
34         @Html.Label("Historia:", "Historia:" + ViewBag.Count, new { @class = "control-label" })
35     </div>
36
37     <div class="form-group">
38         @Html.ActionLink(
39             "Back",                                // linkText
40             "Back",                                // actionName
41             Model,
42             new { @class = "btn btn-primary btn-large" }
43         )
44         <button type="submit" class="btn btn-primary">Policz</button>
45     </div>
46 }
```



Area folder structure

Graphical
User
Interfaces
(EGUI)

Julian Myrcha

Calc Sample

Web servers

HTTP

HTML

document
structure

CSS

Bootstrap

Character
encoding

MVC

Summary

State

state
session state
session example
extension

TempData

TempData

Provider

rzzut ekranu

Startup.cs

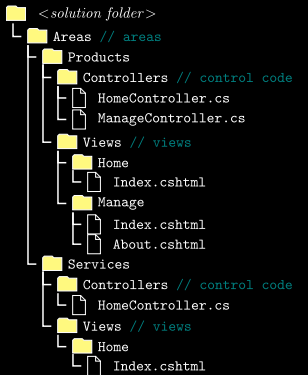
Calc.cs

HomeController.cs

index.cshtml

Area folder
structure

Scaffold



- 1 `dotnet tool install -g dotnet-aspnet-codegenerator`
- 2 `dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design`
- 3 `dotnet-aspnet-codegenerator area Products`
- 4 `dotnet add package Microsoft.EntityFrameworkCore.Tools`
- 5 `dotnet add package Microsoft.EntityFrameworkCore.Sqlite`



Scaffold

write down model class

```
1 using System.ComponentModel.DataAnnotations;
2
3 namespace sample.Models;
4
5 public class Employee
6 {
7     public int EmployeeId { get; set; }
8     public string? FirstName { get; set; }
9     public string? LastName { get; set; }
10    [DataType(DataType.Date)]
11    public DateTime BirthDate { get; set; }
12    public decimal Salary { get; set; }
13 }
```





Scaffold

run scaffolding

```
1 dotnet aspnet-codegenerator controller -name EmployeeController -m Employee \
2 -dc sample.Data.EmployeeContext --relativeFolderPath Controllers \
3 --useDefaultLayout --referenceScriptLibraries --databaseProvider sqlite
4 Building project ...
5 Finding the generator 'controller'...
6 Running the generator 'controller'...
7 Minimal hosting scenario!
8 Generating a new DbContext class 'sample.Data.EmployeeContext'
9 Attempting to compile the application in memory with the added DbContext.
10 Attempting to figure out the EntityFramework metadata for the model and DbContext: 'Employee'
11 Using database provider 'Microsoft.EntityFrameworkCore.Sqlite'!
12 Added DbContext : '/Data/EmployeeContext.cs'
13 Added Controller : '/Controllers/EmployeeController.cs'.
14 Added View : /Views/Employee/Create.cshtml
15 Added View : /Views/Employee/Edit.cshtml
16 Added View : /Views/Employee/Details.cshtml
17 Added View : /Views/Employee/Delete.cshtml
18 Added View : /Views/Employee/Index.cshtml
```



Scaffold

- created Context
- created controller
Controllers/
EmployeeCon-
troller
 - Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Threading.Tasks;
5 using Microsoft.EntityFrameworkCore;
6 using sample.Models;
7 namespace sample.Data {
8     public class EmployeeContext : DbContext {
9         public EmployeeContext
10             (DbContextOptions<EmployeeContext> options)
11             : base(options) {
12         }
13
14         public DbSet<sample.Models.Employee>
15             Employee { get; set; } = default!;
16     }
17 }
```



Scaffold

- created Context
 - created controller
- ## Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 namespace sample.Controllers {  
2     public class EmployeeController : Controller {  
3         private readonly EmployeeContext _context;  
4         public EmployeeController(EmployeeContext context) ...  
5         public async Task<IActionResult> Index() ...  
6         public async Task<IActionResult> Details(int? id)  
7         public IActionResult Create() ...  
8         [HttpPost] public async Task<IActionResult> Create(  
9             Employee employee) ...  
10        public async Task<IActionResult> Edit(int? id) ...  
11        [HttpPost] public async Task<IActionResult> Edit(int id,  
12            Employee employee) ...  
13        public async Task<IActionResult> Delete(int? id) ...  
14        [HttpPost, ActionName("Delete")]  
15        public async Task<IActionResult> DeleteConfirmed(int id)  
16        private bool EmployeeExists(int id) ...  
17    }  
18 }
```



Scaffold

- created Context
 - created controller
- ### Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 // GET: Employee
2 public async Task<IActionResult> Index()
3 {
4     return _context.Employee != null ?
5     View(await _context.Employee.ToListAsync()) :
6     Problem("Entity set 'EmployeeContext.Employee' is null.");
7 }
```



Scaffold

- created Context
 - created controller
- ### Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET] Create
 - [POST] Create
 - [GET] Edit
 - [POST] Edit
 - [GET] Delete
 - [POST] Delete

```
1 // GET: Employee/Details/5
2 public async Task<IActionResult> Details(int? id)
3 {
4     if (id == null || _context.Employee == null)
5     {
6         return NotFound();
7     }
8
9     var employee = await _context.Employee
10 .FirstOrDefaultAsync(m => m.EmployeeId == id);
11     if (employee == null)
12     {
13         return NotFound();
14     }
15
16     return View(employee);
17 }
```



Scaffold

- created Context
 - created controller
- Controllers/
EmployeeCon-
troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 // GET: Employee/Create
2 public IActionResult Create()
3 {
4     return View();
5 }
```



Scaffold

- created Context
- created controller
Controllers/
EmployeeCon-
troller
 - Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 // POST: Employee/Create
2 // To protect from overposting attacks,
3 // enable the specific properties you want to bind to.
4 // http://go.microsoft.com/fwlink/?LinkId=317598.
5 [HttpPost]
6 [ValidateAntiForgeryToken]
7 public async Task<IActionResult> Create(
8     [Bind("EmployeeId,FirstName,LastName,BirthDate,Salary")]
9     Employee employee)
10 {
11     if (ModelState.IsValid)
12     {
13         _context.Add(employee);
14         await _context.SaveChangesAsync();
15         return RedirectToAction(nameof(Index));
16     }
17     return View(employee);
18 }
```



Scaffold

- created Context
 - created controller
- ## Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET>Delete
 - [POST>Delete

```
1 // GET: Employee/Edit/5
2 public async Task<IActionResult> Edit(int? id)
3 {
4     if (id == null || _context.Employee == null)
5     {
6         return NotFound();
7     }
8
9     var employee = await _context.Employee.FindAsync(id);
10    if (employee == null)
11    {
12        return NotFound();
13    }
14    return View(employee);
15 }
```




Scaffold

- created Context
 - created controller
- ### Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 // POST: Employee/Edit/5
2 [HttpPost]
3 [ValidateAntiForgeryToken]
4 public async Task<IActionResult> Edit(int id,
5     [Bind("EmployeeId,FirstName,LastName,BirthDate,Salary")]
6         Employee employee){
7     if (id != employee.EmployeeId) {
8         return NotFound();
9     }
10
11     if (ModelState.IsValid) {
12         try {
13             _context.Update(employee);
14             await _context.SaveChangesAsync();
15         }
16         catch (DbUpdateConcurrencyException) {
17             if (!EmployeeExists(employee.EmployeeId)) {
18                 return NotFound();
19             }
20             else {
21                 throw;
22             }
23         }
24         return RedirectToAction(nameof(Index));
25     }
26     return View(employee);
27 }
```



Scaffold

- created Context
 - created controller
- ### Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 public async Task<IActionResult> Delete(int? id)
2 {
3     if (id == null || _context.Employee == null)
4     {
5         return NotFound();
6     }
7
8     var employee = await _context.Employee
9         .FirstOrDefaultAsync(m => m.EmployeeId == id);
10    if (employee == null)
11    {
12        return NotFound();
13    }
14
15    return View(employee);
16 }
```



Scaffold

- created Context
 - created controller
- ## Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 // POST: Employee/Delete/5
2 [HttpPost, ActionName("Delete")]
3 [ValidateAntiForgeryToken]
4 public async Task<IActionResult> DeleteConfirmed(int id)
5 {
6     if (_context.Employee == null)
7     {
8         return Problem("Entity set 'EmployeeContext.Employee' is null.");
9     }
10    var employee = await _context.Employee.FindAsync(id);
11    if (employee != null)
12    {
13        _context.Employee.Remove(employee);
14    }
15
16    await _context.SaveChangesAsync();
17    return RedirectToAction(nameof(Index));
18 }
```



Scaffold

- created Context
- created controller
**Controllers/
EmployeeCon-
troller**
 - Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
1 // Controllers/EmployeeController
2 using System;
3 using System.Collections.Generic;
4 using System.Linq;
5 using System.Threading.Tasks;
6 using Microsoft.AspNetCore.Mvc;
7 using Microsoft.AspNetCore.Mvc.Rendering;
8 using Microsoft.EntityFrameworkCore;
9 using sample.Data;
10 using sample.Models;
11
12 namespace sample.Controllers
13 {
14     public class EmployeeController : Controller
15     {
16         readonly EmployeeContext _context;
17
18         public EmployeeController(EmployeeContext context)
19         {
20             _context = context;
21         }
22
23         // GET: Employee
24         public async Task<IActionResult> Index()
25         {
26             return _context.Employee != null ?
27                 View(await _context.Employee.ToListAsync()) :
28                 Problem("Entity set 'EmployeeContext.Employee' is null.");
29         }
30
31         // GET: Employee/Details/5
32         public async Task<IActionResult> Details(int? id)
33         {
34             if (id == null || _context.Employee == null)
```



Scaffold

- created Context
- created controller
Controllers/
EmployeeCon-
troller
 - Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
35 {
36     return NotFound();
37 }
38
39 var employee = await _context.Employee
40     .FirstOrDefaultAsync(m => m.EmployeeId == id);
41 if (employee == null)
42 {
43     return NotFound();
44 }
45
46     return View(employee);
47 }
48
49 // GET: Employee/Create
50 public IActionResult Create()
51 {
52     return View();
53 }
54
55 // POST: Employee/Create
56 // To protect from overposting attacks, enable the specific
57 // properties you want to bind to.
58 // For more details, see
59 // http://go.microsoft.com/fwlink/?LinkId=317598.
60 [HttpPost]
61 [ValidateAntiForgeryToken]
62 public async Task<ActionResult>
63     Create([Bind("EmployeeId,FirstName,LastName,BirthDate,Salary")]
64         Employee employee)
65 {
66     if (ModelState.IsValid)
67     {
68         _context.Add(employee);
69         await _context.SaveChangesAsync();
70         return RedirectToAction(nameof(Index));
71     }
72     return View(employee);
73 }
```



Scaffold

- created Context
 - created controller
- ## Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
70
71 // GET: Employee/Edit/5
72 public async Task<IActionResult> Edit(int? id)
73 {
74     if (id == null || _context.Employee == null)
75     {
76         return NotFound();
77     }
78
79     var employee = await _context.Employee.FindAsync(id);
80     if (employee == null)
81     {
82         return NotFound();
83     }
84     return View(employee);
85 }
86
87 // POST: Employee/Edit/5
88 // To protect from overposting attacks,
89 // enable the specific properties you want to bind to.
90 // For more details, see
91 // http://go.microsoft.com/fwlink/?LinkId=317598.
92 [HttpPost]
93 [ValidateAntiForgeryToken]
94 public async Task<IActionResult> Edit(int id,
95     [Bind("EmployeeId,FirstName,LastName,BirthDate,Salary")] Employee
96     employee)
97 {
98     if (id != employee.EmployeeId)
99     {
100         return NotFound();
101     }
102     if (ModelState.IsValid)
103     {
104         try
```



Scaffold

- created Context
 - created controller
- ## Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
105     _context.Update(employee);
106     await _context.SaveChangesAsync();
107 }
108 catch (DbUpdateConcurrencyException)
109 {
110     if (!EmployeeExists(employee.EmployeeId))
111     {
112         return NotFound();
113     }
114     else
115     {
116         throw;
117     }
118 }
119 return RedirectToAction(nameof(Index));
120 }
121 return View(employee);
122 }
123
124 // GET: Employee/Delete/5
125 public async Task<IActionResult> Delete(int? id)
126 {
127     if (id == null || _context.Employee == null)
128     {
129         return NotFound();
130     }
131
132     var employee = await _context.Employee
133         .FirstOrDefaultAsync(m => m.EmployeeId == id);
134     if (employee == null)
135     {
136         return NotFound();
137     }
138
139     return View(employee);
```



Scaffold

- created Context
 - created controller
- ## Controllers/ EmployeeCon- troller
- Index
 - Details
 - [GET]Create
 - [POST]Create
 - [GET]Edit
 - [POST]Edit
 - [GET]Delete
 - [POST]Delete

```
140 }
141
142 // POST: Employee/Delete/5
143 [HttpPost, ActionName("Delete")]
144 [ValidateAntiForgeryToken]
145 public async Task<IActionResult> DeleteConfirmed(int id)
146 {
147     if (_context.Employee == null)
148     {
149         return Problem("Entity set 'EmployeeContext.Employee' is null.");
150     }
151     var employee = await _context.Employee.FindAsync(id);
152     if (employee != null)
153     {
154         _context.Employee.Remove(employee);
155     }
156
157     await _context.SaveChangesAsync();
158     return RedirectToAction(nameof(Index));
159 }
160
161 private bool EmployeeExists(int id)
162 {
163     return (_context.Employee?.Any(e => e.EmployeeId ==
164         id)).GetValueOrDefault();
165 }
166 }
```




Scaffold Edit view

```
1 // Models/Employee.cs
2 using System.ComponentModel.DataAnnotations;
3
4 namespace sample.Models;
5
6 public class Employee
7 {
8     public int EmployeeId { get; set; }
9     public string? FirstName { get; set; }
10    public string? LastName { get; set; }
11    [DataType(DataType.Date)]
12    public DateTime BirthDate { get; set; }
13    public decimal Salary { get; set; }
14 }
```

```
1 dotnet aspnet-codegenerator view EditEmployee Edit -m Employee -outDir Views/Employee
↩ --referenceScriptLibraries --databaseProvider sqlite
2 Building project ...
3 Finding the generator 'view'...
4 Running the generator 'view'...
5 Added View : /Views/Employee/EditEmployee.cshtml
6 RunTime 00:00:06.18
```