

Build web apps with



by learning from a learner

Rey Valeza

Introduction

This is a book written by a learner for fellow learners of Vibe.d. I am no expert on D, but I am happy to note that an expert-level knowledge of D is *not* required to develop apps in Vibe.d.

I hankered for Vibe.d and its simplicity after looking at the complexity of Spring / Hibernate and the technologies needed just to build a simple application. Yes, Java is fantastic. Fantastically complex. I long for the simpler days of Java up to JDK 1.4 with just JSP's, servlets and JDBC.

I built a web app with Vibe.d a few years back. I found there aren't any drastic changes since then, and I'm glad. That is how it should be. That is why PHP is still so popular: simplicity and no drastic changes. Young people can pick up any old book or tutorial and find it isn't much different from the current PHP.

Why did Rails fail to gain traction? Because by the time it was attracting droves of potential converts, they changed drastically by adopting the then hot-off-the-oven REST, ignoring the users who were already happy with the previous ways and frustrating new converts desperately looking for updated tutorials and documentation.

Why did Meteor fail too? Because by the time it was catching people's curiosity, they decided to adopt the very popular React. For existing users, the quandary was to go the old Blaze way or the React way. Which way will win eventually? Since there was no way of divining the future, new and old users alike took the third way: abandon ship.

You do not make drastic changes to your already revolutionary product when you haven't achieved a critical mass of users yet. You will not expand user base by constantly adopting each and every sizzling-new buzzword, you only end up losing them.

Remember how Java did it? There was just so much hype at the beginning. There was just so much buzz. Java was everywhere. Everybody was curious. Converts came in droves. And Sun backed it all up with solid, updated documentation. That's how you do it. The Sun people were pros compared to these new upstarts. So even if Java kept changing, it has already achieved a critical mass of followers long ago so that users have no choice but to follow changes as a captive audience. Java slew the lumbering dinosaurs then. Sadly, Java is now the dinosaur.

The Vibe.d framework was created by Sönke Ludwig out of frustration with existing frameworks, notably Node.js. If you are intrigued by that, find out more at <https://vibed.org/>. You can find the forum at <https://forum.rejectedsoftware.com/>.

Vibe.d doesn't rely on hype, which plays against it. I feel there is a dearth of tutorials on Vibe.d. After opting to learn Vibe.d again, I found I forgot so much, so I decided to write this book while I am re-learning it.

Configuring your Windows system for Vibe.d

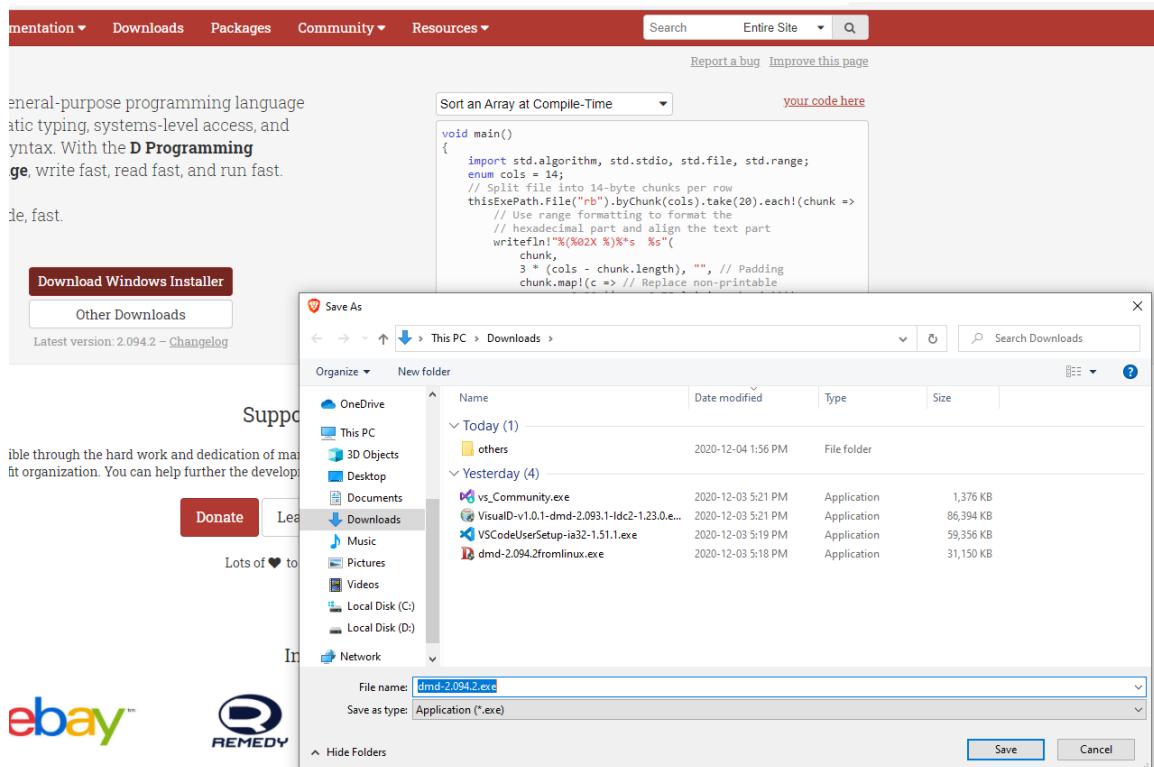
I am using Windows 10 so I hope your system mimics mine, but you should be fine with another OS you are familiar with, and I assume you know the equivalent Windows commands in your system's shell terminal. We will be using the terminal or command window a lot. I often find that Linux users easily follow Windows-based tutorials but not the other way around. Which means most Linux users are or were Windows users too.

I did the projects in an Ubuntu system before writing this book, so I rewrote the projects in Windows 10 as I know most developers use or prefer Windows.

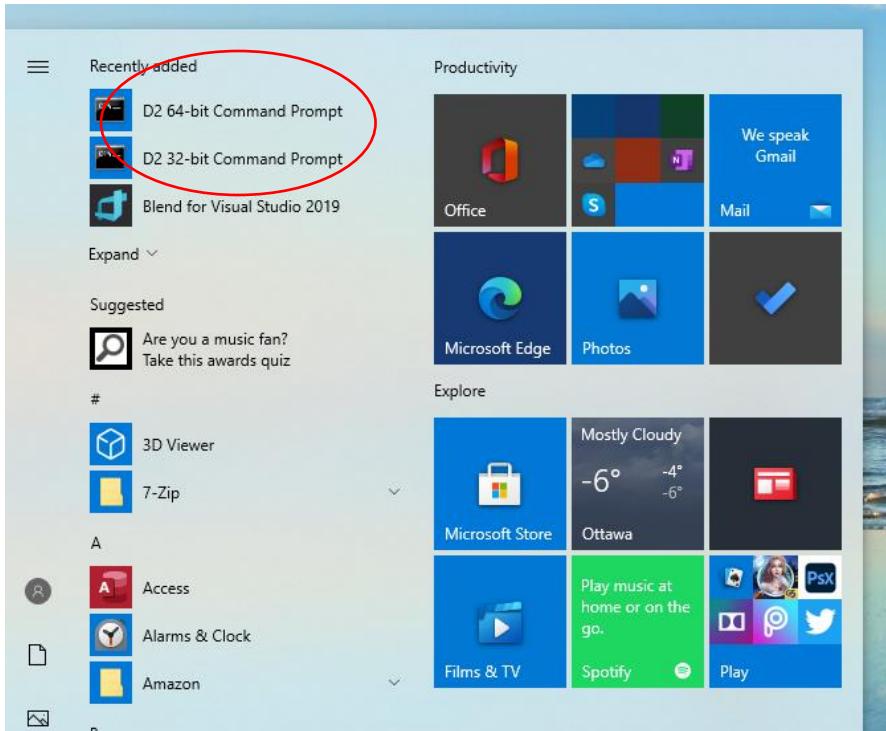
D is a language that compiles to native code. The Vibe.d framework is essentially a set of libraries written in D, which means your code needs to be linked to those libraries and dependencies and compiled into a native executable. To develop applications in Vibe.d, we need at least a D compiler and a text editor.

Regarding compilers, there are three choices: DMD, GDC and LDC. They tell us that for development purposes, we should use the DMD compiler for its speed of compilation. When we are ready to deploy the final production version, then we should use one of the other compilers (GDC or LDC) for the runtime speed and optimizations of the binary output. This book uses the DMD compiler. I hope you do too.

Go to <https://dlang.org/> and click on the ‘Download Windows Installer’ button.



After installation, you should have new entries in your start menu.



Go ahead and open a terminal or command window. We will be using that window a lot.

To test the installation of the DMD compiler, type `dmd` on the terminal window. If the installation is successful, it will fill the window with help information.

```
C:\DCompiler>dmd
DMD64 D Compiler v2.094.2-dirty
Copyright (C) 1999-2020 by The D Language Foundation, All Rights Reserved written by
Walter Bright
```

Documentation: <https://dlang.org/>
Config file: C:\DCompiler\dmd2\windows\bin64\sc.ini
Usage:
`dmd [<option>...] <file>...`
`dmd [<option>...] -run <file> [<arg>...]`

Where:

`<file>` D source file
`<arg>` Argument to pass when running the resulting program

`<option>:`

`@<cmdfile>` read arguments from cmdfile
`-allinst` generate code for all template instantiations

- betterC omit generating some runtime information and helper functions
- boundscheck=[on|safeonly|off] bounds checks on, in @safe only, or off
- c compile only, do not link
- check=[assert|bounds|in|invariant|out|switch][=[on|off]] enable or disable specific checks
- check=[h|help|?] list information on all available checks
- checkaction=[D|C|halt|context] behavior on assert/boundscheck/finalswitch failure
- checkaction=[h|help|?] list information on all available check actions
- color turn colored console output on
- color=[on|off|auto] force colored console output on or off, or only when not redirected (default)
- conf=<filename> use config file at filename
- cov do code coverage analysis
- cov=ctfe Include code executed during CTFE in coverage report
- cov=<nnn> require at least nnn% code coverage
- D generate documentation
- Dd<directory> write documentation file to directory
- Df<filename> write documentation file to filename
- d silently allow deprecated features and symbols
- de issue an error when deprecated features or symbols are used (halt compilation)
- dw issue a message when deprecated features or symbols are used (default)
- debug compile in debug code
- debug=<level> compile in debug code <= level
- debug=<ident> compile in debug code identified by ident
- debuglib=<name> set symbolic debug library to name
- defaultlib=<name> set default library to name
- deps print module dependencies (imports/file/version/debug/lib)
- deps=<filename> write module dependencies to filename (only imports)
- extern-std=<standard> set C++ name mangling compatibility with <standard>
- extern-std=[h|help|?] list all supported standards
- g add symbolic debug info
- gf emit debug info for all referenced types
- gs always emit stack frame
- gx add stack stomping code
- H generate 'header' file
- Hd=<directory> write 'header' file to directory
- Hf=<filename> write 'header' file to filename

-HC=[silent|verbose]]
 generate C++ 'header' file

-HC=[?|h|help] list available modes for C++ 'header' file generation

-HCd=<directory> write C++ 'header' file to directory

-H Cf=<filename> write C++ 'header' file to filename

--help print help and exit

-I=<directory> look for imports also in directory

-i[=<pattern>] include imported modules in the compilation

-ignore ignore unsupported pragmas

-inline do function inlining

-J=<directory> look for string imports also in directory

-L=<linkerflag> pass linkerflag to link

-lib generate library rather than object files

-lowmem enable garbage collection for the compiler

-m32 generate 32 bit code

-m32mscoff generate 32 bit code and write MS-COFF object files

-m64 generate 64 bit code

-main add default main() (e.g. for unittesting)

-man open web browser on manual page

-map generate linker .map file

-mcpu=<id> generate instructions for architecture identified by 'id'

-mcpu=[h|help|?] list all architecture options

-mixin=<filename> expand and save mixins to file specified by <filename>

-mscrtlib=<libname>
 MS C runtime library to reference from main/WinMain/DllMain

-mv=<package.module>=<filespec>
 use <filespec> as source file for <package.module>

-noboundscheck no array bounds checking (deprecated, use -boundscheck=off)

-O optimize

-o- do not write object file

-od=<directory> write object & library files to directory

-of=<filename> name output file to filename

-op preserve source path for output files

-preview=<name> enable an upcoming language change identified by 'name'

-preview=[h|help|?] list all upcoming language changes

-profile profile runtime performance of generated code

-profile=gc profile runtime allocations

-release compile release version

-revert=<name> revert language change identified by 'name'

-revert=[h|help|?] list all revertable language changes

-run <srcfile> compile, link, and run the program srcfile

-shared generate shared library (DLL)

```
-transition=<name>
    help with language change identified by 'name'
-transition=[h|help|?]
    list all language changes
-unittest      compile in unit tests
-v            verbose
-vcolumns     print character (column) numbers in diagnostics
-verror-style=[digitalmars|gnu]
    set the style for file/line number annotations on compiler messages
-verrors=<num>  limit the number of error messages (0 means unlimited)
-verrors=context  show error messages with the context of the erroring source line
-verrors=spec    show errors from speculative compiles such as __traits(compiles,...)
--version      print compiler version and exit
-version=<level>  compile in version code >= level
-version=<ident>  compile in version code identified by ident
-vgc          list all gc allocations including hidden ones
-vtls         list all variables going into thread local storage
-vtemplates=[list-instances]
    list statistics on template instantiations
-w            warnings as errors (compilation will halt)
-wi           warnings as messages (compilation will continue)
-X            generate JSON file
-Xf=<filename>  write JSON file to filename
```

C:\DCompiler>

These two DMD options are helpful:

dmd --help will display the same information.

dmd --version will show you just the DMD compiler version.

The DMD installer comes with the dub package manager. It is the default package manager for D projects and was created by the author of the Vibe.d framework himself, Sönke Ludwig.

To test dub, simply type in your terminal

dub

If it complains that there is no project manifest, then you are good.

Another way to test it is to type

dub help

or

dub -h

or

dub --help

This will show the dub help information.

To show only the version number, type

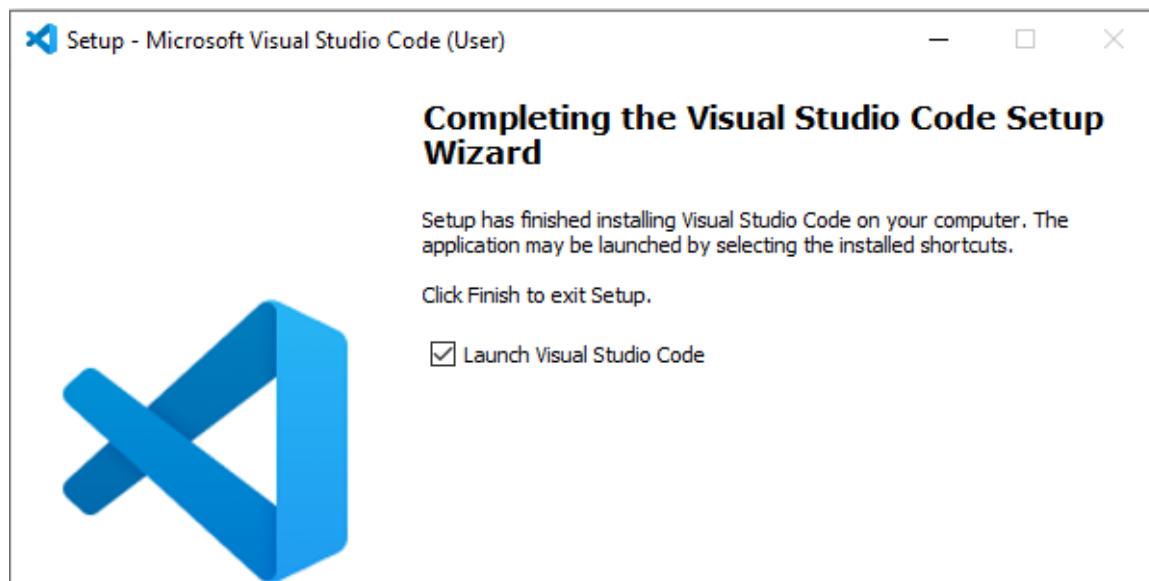
dub --version

After the compiler is installed, the next thing we need is a text editor or an IDE. You can use your favorite IDE/text editor while following this book, but this book will use the ubiquitous and free Visual Studio Code. Since you are using Windows, you must have that already. To those who haven't yet, head to

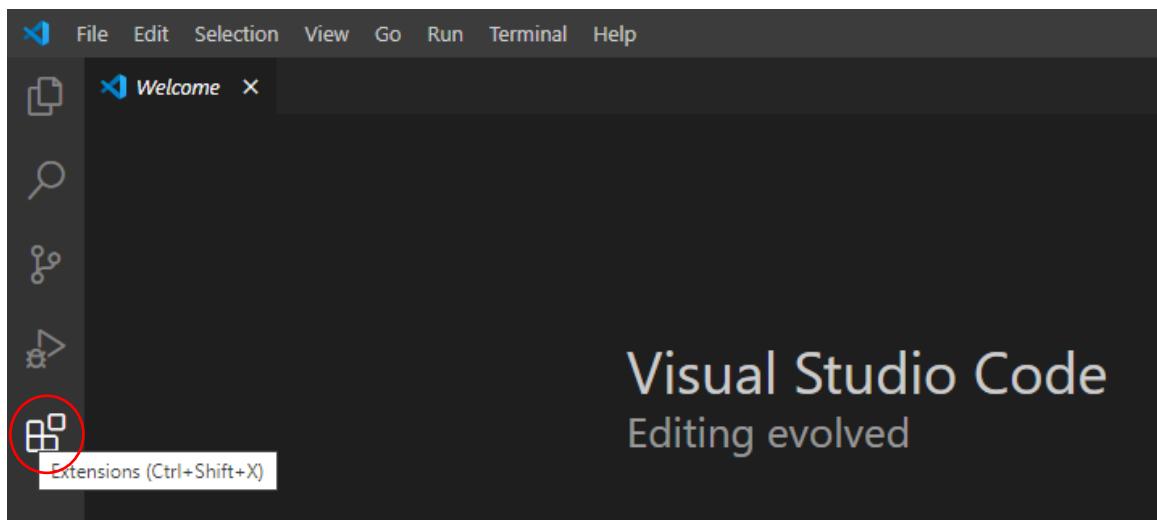
<https://code.visualstudio.com/download>

and choose the installer appropriate for your system.

Simply double-click on the downloaded file and follow the instructions on the setup wizard.



After installation, add the D language extension so VS Code can recognize D language code.



Add the D language utility extension pack. With this pack, you will have features like syntax color highlighting, code-completion and some error flagging.

A screenshot of the Visual Studio Code Extension Marketplace. On the left, there is a sidebar with icons for search, filters, and categories. The main area shows a list of extensions under the heading "D language". One extension, "D Language utility extension pack" by WebFreak, is highlighted with a red oval. The details for this extension are shown on the right: it has 6,591 installs and a 5-star rating. It includes features like GDB, LLDB, mago-mi, DCD, DScanner, dfmt, serve-d, package versioning, and more. Below this, another extension, "dlang-bundle" by pflannery, is listed. It contains "Extension Pack (3)" which includes "Version Lens" and "D Programming Language (code-d)".

Now you are ready to build web apps in Vibe.d.

The default Hello, World! application

In the terminal, go to your folder of choice or create a new one. Mine is named C:\vibeprojects. Inside that folder, create a new application named hello with the command

```
dub init hello --type=vibe.d
```

or

```
dub init hello -t vibe.d
```

Press enter at every prompt that follows to accept the defaults.

```
C:\vibeprojects> dub init hello -t vibe.d
Package recipe format (sdl/json) [json]:
Name [hello]:
Description [A simple vibe.d server application.]:
Author name [lenovo pc]:
License [proprietary]:
Copyright string [Copyright © 2020, lenovo pc]:
Add dependency (leave empty to skip) []:
Successfully created an empty project in 'C:\vibeprojects\hello'.
Package successfully created in hello
```

Go inside the newly created hello project folder.

```
PS C:\vibeprojects> cd hello
PS C:\vibeprojects\hello> dir
Directory: C:\vibeprojects\hello
```

Mode	LastWriteTime	Length	Name
---	-----	----	---
d----	2020-12-04 8:19 PM		public
d----	2020-12-04 8:19 PM		source
d----	2020-12-04 8:19 PM		views
-a----	2020-12-04 8:19 PM	125	.gitignore
-a----	2020-12-04 8:19 PM	223	dub.json

```
PS C:\vibeprojects\hello>
```

You are now inside the root directory of the hello application. To compile and run this application in one step (you haven't done anything yet!), type

dub

Since this is the first time you run this command, this will download all the dependencies, then compile, then link, then finally run the application.

```
PS C:\vibeprojects\hello> dub
Fetching vibe-core 1.11.1 (getting selected version)...
Fetching botan-math 1.0.3 (getting selected version)...
Fetching taggedalgebraic 0.11.18 (getting selected version)...
Fetching vibe-d 0.9.2 (getting selected version)...
Fetching memutils 1.0.4 (getting selected version)...
Fetching stdx-allocator 2.77.5 (getting selected version)...
Fetching botan 1.12.18 (getting selected version)...
Fetching diet-ng 1.7.4 (getting selected version)...
Fetching openssl 1.1.6+1.0.1g (getting selected version)...
Fetching eventcore 0.9.11 (getting selected version)...
Fetching mir-linux-kernel 1.0.1 (getting selected version)...
Fetching libasync 0.8.6 (getting selected version)...
Performing "debug" build using C:\DCompiler\dmd2\windows\bin\dmd.exe for x86_64.
taggedalgebraic 0.11.18: building configuration "library"...
eventcore 0.9.11: building configuration "winapi"...
stdx-allocator 2.77.5: building configuration "library"...
vibe-core 1.11.1: building configuration "winapi"...
vibe-d:utils 0.9.2: building configuration "library"...
vibe-d:data 0.9.2: building configuration "library"...
mir-linux-kernel 1.0.1: building configuration "library"...
vibe-d:crypto 0.9.2: building configuration "library"...
diet-ng 1.7.4: building configuration "library"...
vibe-d:stream 0.9.2: building configuration "library"...
vibe-d:textfilter 0.9.2: building configuration "library"...
vibe-d:inet 0.9.2: building configuration "library"...
vibe-d:tls 0.9.2: building configuration "openssl-mscoff"...
vibe-d:http 0.9.2: building configuration "library"...
C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\range\primitives.d(175,38):
Deprecation: alias byKeyValue this is deprecated - Iterate over .byKeyValue instead.
C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\range\primitives.d(177,27):
Deprecation: alias byKeyValue this is deprecated - Iterate over .byKeyValue instead.
C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\range\primitives.d(175,38):
Deprecation: alias byKeyValue this is deprecated - Iterate over .byKeyValue instead.
C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\range\primitives.d(177,27):
Deprecation: alias byKeyValue this is deprecated - Iterate over .byKeyValue instead.
vibe-d:mail 0.9.2: building configuration "library"...
vibe-d:mongodb 0.9.2: building configuration "library"...
```

C:\DCompiler\dmd2\windows\bin\..\..\src\druntime\import\core\internal\traits.d(341,61): Deprecation: function std.typecons.Nullable!int.Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\format.d(3648,26): Deprecation: function std.typecons.Nullable!string.Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\format.d(3648,26): Deprecation: function std.typecons.Nullable!(Alternate).Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\format.d(3648,26): Deprecation: function std.typecons.Nullable!(MaxVariable).Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\format.d(3648,26): Deprecation: function std.typecons.Nullable!string.Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\format.d(3648,26): Deprecation: function std.typecons.Nullable!(Alternate).Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

C:\DCompiler\dmd2\windows\bin\..\..\src\phobos\std\format.d(3648,26): Deprecation: function std.typecons.Nullable!(MaxVariable).Nullable.get_ is deprecated - Implicit conversion with alias Nullable.get this will be removed after 2.096. Please use .get explicitly.

vibe-d:redis 0.9.2: building configuration "library"...

vibe-d:web 0.9.2: building configuration "library"...

vibe-d 0.9.2: building configuration "vibe-core"...

hello ~master: building configuration "application"...

Linking...

Copying files for vibe-d:tls...

Running .\hello.exe

[main(----) INF] Listening for requests on http://[::1]:8080/

[main(----) INF] Listening for requests on http://127.0.0.1:8080/

[main(----) INF] Please open http://127.0.0.1:8080/ in your browser.

The last three lines indicate that the project has successfully compiled and your server is now running.

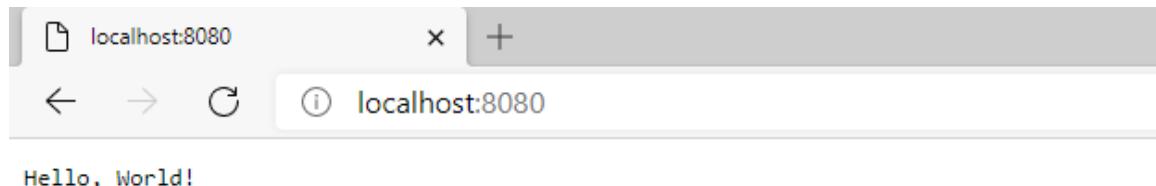
Open your browser and point it to localhost, port 8080.

<http://127.0.0.1:8080>

or

<http://localhost:8080>

and you will see the ‘Hello, World!’ message in your browser.



To stop the running application, press Ctrl-C (Control-C) on the terminal window.

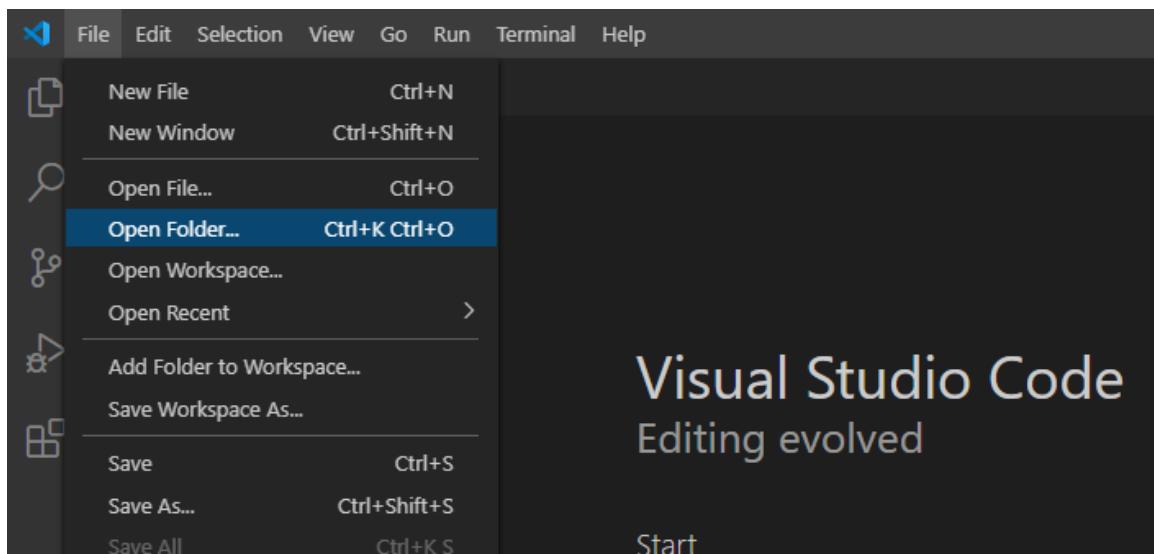
```
[00000000(----) INF] Received signal 2. Shutting down.  
Warning: 6 socket handles leaked at driver shutdown.  
Warning: 6 socket handles leaked at driver shutdown.  
^C  
c:\vibeprojects\hello>
```

The common commands for dub are

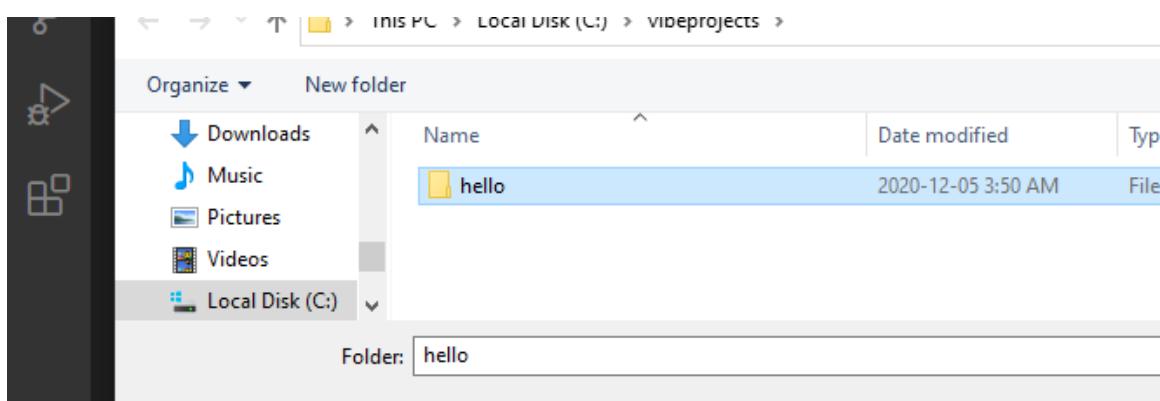
dub init	- create a new project
dub run	- build and run combined, ignoring unit tests
dub	- same as dub run
dub build	- completely compile the whole project and its dependencies
dub test	- run the unit tests of the application

Let us take a closer look at the default application.

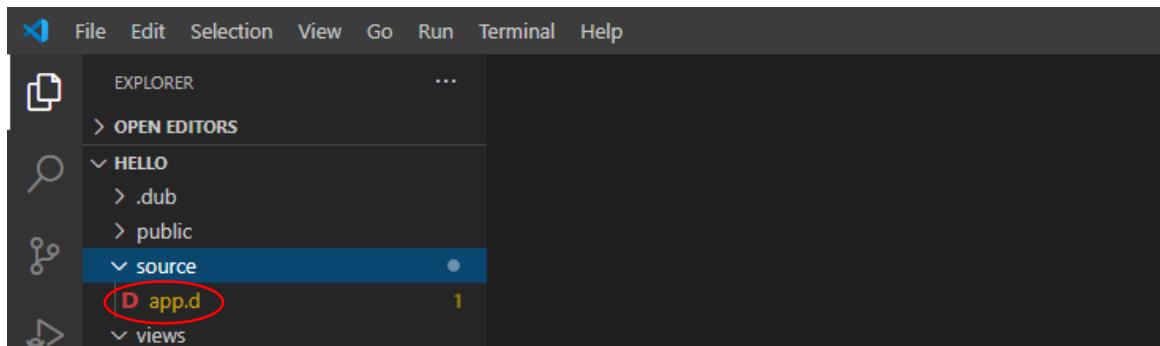
Open VS Code and go to File->Open folder...



Then open the new application folder hello.



In the Explorer window of VS Code, open source\app.d.



And you will see the automatically generated code.

```
import vibe.vibe;

void main()
```

```
{  
    auto settings = new HTTPServerSettings;  
    settings.port = 8080;  
    settings.bindAddresses = [":1", "127.0.0.1"];  
    listenHTTP(settings, &hello);  
  
    logInfo("Please open http://127.0.0.1:8080/ in your browser.");  
    runApplication();  
}  
  
void hello(HTTPServerRequest req, HTTPServerResponse res)  
{  
    res.writeBody("Hello, World!");  
}
```

In the main() method of source\app.d,

- a new HTTPServerSettings object is created
- the HTTP port number is set to 8080 (which you can change)
- the IP address that the server will listen to is set to localhost in both IPv6 (“::1”) and IPv4 (“127.0.0.1”) format
- the listenHTTP(settings, &hello) call means ‘run the hello() function using these settings’.
- the runApplication() call starts the ball rolling

Hello, main()!

D is a case-sensitive language, just like C, C++, Java and C#.

All console applications in D start with a `main()` function. It is the entry point of the application.

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
```

The line

```
import vibe.vibe;
```

means import the file (or module) `vibe\vibe.d`. The convention in D is to use the file name as the module name so keeping track of namespaces and files are simpler. The `vibe.vibe` library module contains almost all of the Vibe.d API so it is very convenient for users.

The `import` directive is equivalent to C and C++'s `#include` preprocessor directive, but C and C++ insert the whole file into the importing file while D does not.

```
void main()
```

The return value of `main()` can either be `void` or `int`, which is the exit status. The `main()` function can also take an array of strings, but we don't use them in Vibe.d.

```
auto settings = new HTTPServerSettings;
```

That line could have been written as

```
HTTPServerSettings settings = new HTTPServerSettings();
```

The term `auto` means let the compiler deduce the datatype from the context, so it's another convenience to us from D. Writing `auto` is a lot simpler than writing `HTTPServerSettings`.

Parentheses are optional in D function calls when there are no arguments.

Using your own HTML page

Let us create a new file named views\helloagain.html. Stop the application if it is still running by pressing Ctrl-C in the terminal window (twice if need be).

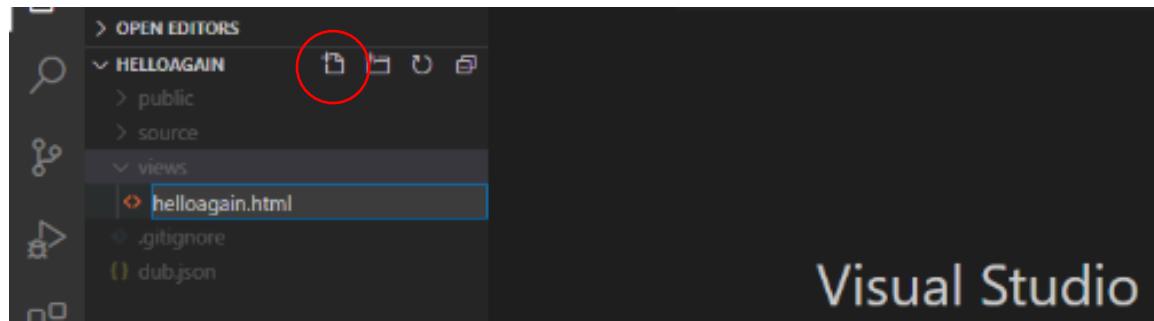
^C

c:\vibeprojects\hello>^D[00000000(----) INF] Received signal 2. Shutting down.

Warning: 4 socket handles leaked at driver shutdown.

Warning: 4 socket handles leaked at driver shutdown.

In the Explorer window of VS Code, create a new HTML file inside the views\ folder and name it helloagain.html.



And type the following in views\helloagain.html.

```
<html>
  <head>
    <title>Hello again!</title>
  </head>
  <body>
    <h2>Hello again, World!</h2>
  </body>
</html>
```

Vibe.d is a stickler for proper indentation of HTML tags, so follow proper indentation.

Next, edit source\app.d to make it display the helloagain.html page.

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];
```

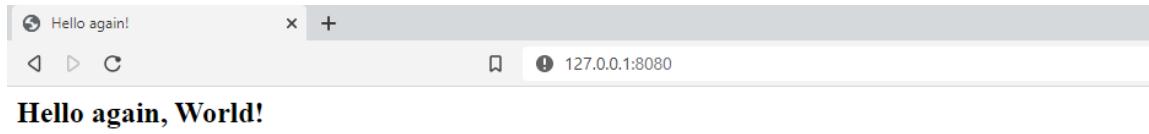
```
listenHTTP(settings, staticTemplate!"helloagain.html");

logInfo("Please open http://127.0.0.1:8080/ in your browser.");
runApplication();
}
```

And delete the hello() function because it is no longer called. Compile and run the project with dub.

```
c:\vibeprojects\hello>dub
Performing "debug" build using C:\DCompiler\dmd2\windows\bin\dmd.exe for x86_64.
taggedalgebraic 0.11.18: target for configuration "library" is up to date.
eventcore 0.9.11: target for configuration "winapi" is up to date.
stdx-allocator 2.77.5: target for configuration "library" is up to date.
vibe-core 1.11.1: target for configuration "winapi" is up to date.
vibe-d:utils 0.9.2: target for configuration "library" is up to date.
vibe-d:data 0.9.2: target for configuration "library" is up to date.
mir-linux-kernel 1.0.1: target for configuration "library" is up to date.
vibe-d:crypto 0.9.2: target for configuration "library" is up to date.
diet-ng 1.7.4: target for configuration "library" is up to date.
vibe-d:stream 0.9.2: target for configuration "library" is up to date.
vibe-d:textfilter 0.9.2: target for configuration "library" is up to date.
vibe-d:inet 0.9.2: target for configuration "library" is up to date.
vibe-d:tls 0.9.2: target for configuration "openssl-mscoff" is up to date.
vibe-d:http 0.9.2: target for configuration "library" is up to date.
vibe-d:mail 0.9.2: target for configuration "library" is up to date.
vibe-d:mongodb 0.9.2: target for configuration "library" is up to date.
vibe-d:redis 0.9.2: target for configuration "library" is up to date.
vibe-d:web 0.9.2: target for configuration "library" is up to date.
vibe-d 0.9.2: target for configuration "vibe-core" is up to date.
helloagain ~master: building configuration "application"...
Compiling Diet HTML template helloagain.html...
Linking...
To force a rebuild of up-to-date targets, run again with --force.
Copying files for vibe-d:tls...
Running .\hello.exe
[main(---) INF] Listening for requests on http://[::1]:8080/
[main(---) INF] Listening for requests on http://127.0.0.1:8080/
[main(---) INF] Please open http://127.0.0.1:8080/ in your browser.
```

You see that the required libraries were no longer downloaded so the compilation is a bit faster. Refresh your browser and you should see the new message.



To stop the running app, press Ctrl-C. Twice if needed.

As you might have noticed, this is the conventional directory layout of a Vibe.d app using dub:

\source	- your D code should reside here
\views	- the HTML or template pages stay here
\public	- for other files such as CSS, scripts, images, videos, etc.

Serving files other than HTML

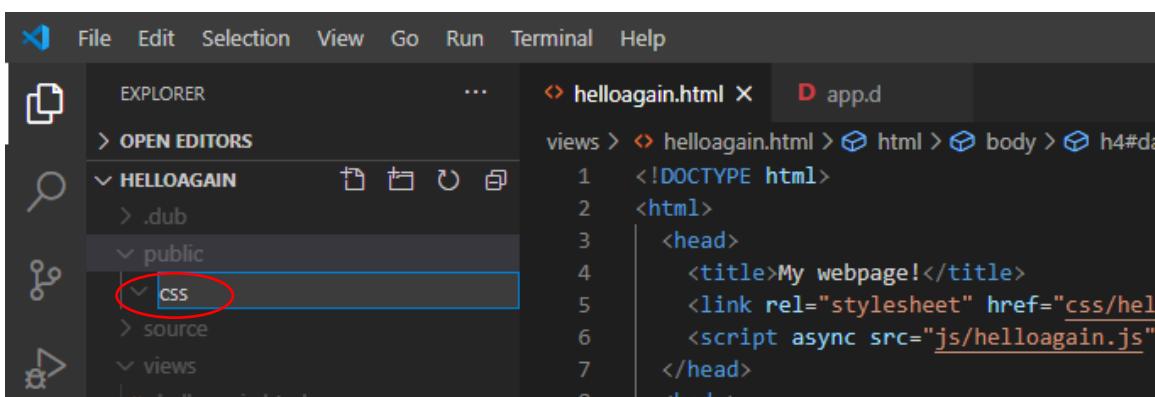
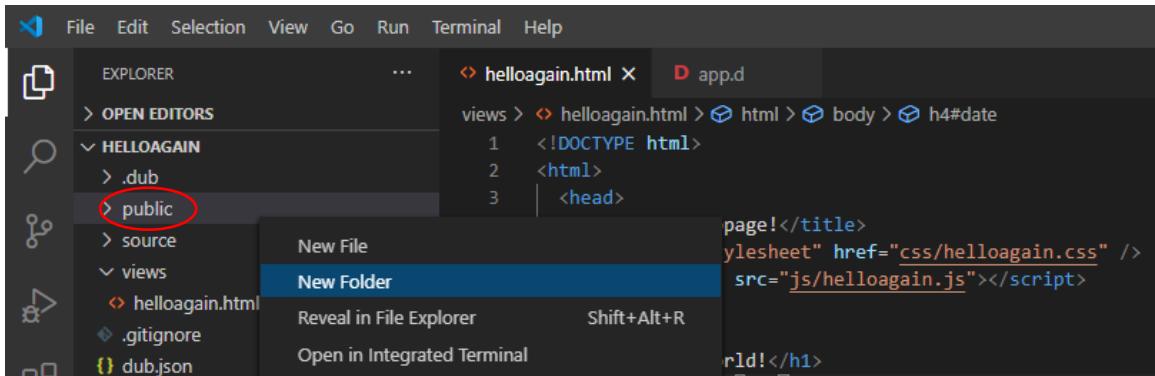
What if an HTML page requires CSS and JavaScript files? Where do we put them?
Answer: in the \public\ folder.

Edit views\helloagain.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>My webpage!</title>
    <link rel="stylesheet" href="css/helloagain.css" />
    <script async src="js/helloagain.js"></script>
  </head>
  <body>
    <h1>Hello, World!</h1>
    <h4 id='date'></h4>
    <div class="image-section">
      <div class="section-style">
        
        <p>A random image courtesy of unsplash.com.</p>
      </div>
      <div class="section-style">
        
        <p>A random image courtesy of unsplash.com.</p>
      </div>
    </div>
    <div class="image-section">
      <div class="section-style">
        
        <p>A random image courtesy of unsplash.com.</p>
      </div>
      <div class="section-style">
        
        <p>A random image courtesy of unsplash.com.</p>
      </div>
    </div>
  </body>
</html>
```

Actually, I got this from the wild wild web courtesy of Programming Liftoff
(<https://dev.to/programliftoff/create-a-basic-webpage-with-css-and-javascript--104i>)

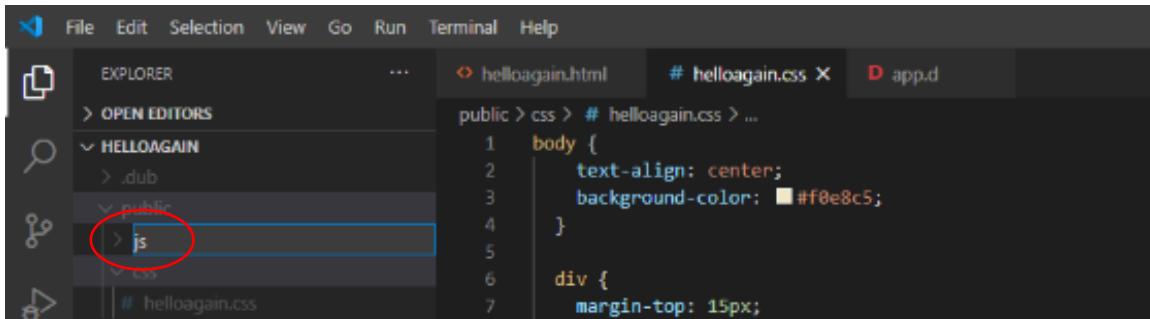
Next, create a css\ folder inside the public\ folder (you can right-click on it)



Then create the public\css\helloagain.css stylesheet file (also from Programming Liftoff) inside that new folder.

```
body {  
    text-align: center;  
    background-color: #f0e8c5;  
}  
  
div {  
    margin-top: 15px;  
}  
  
.image-section {  
    display: flex;  
    justify-content: center;  
}  
  
.section-style {  
    margin-right: 25px;  
    margin-left: 25px;  
    background-color: white;  
}
```

Next, create the \js folder, also inside the public\ folder.



Then create the public\js\helloagain.js with the following contents (also from Programming Liftoff)

```
document.getElementById('date').innerHTML = new Date().toLocaleString();
```

Then edit source\app.d to use a router

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    auto router = new URLRouter;
    router.get("/", staticTemplate!"helloagain.html");
    router.get("*", serveStaticFiles("public/"));

    listenHTTP(settings, router);

    runApplication();
}
```

The line

```
router.get("/", staticTemplate!"helloagain.html");
```

means to display helloagain.html when the web browser URL shows "/", meaning the helloagain.html is the index page or home page. The "/" URL expands to <http://localhost:8080/>, which is the root of the application.

The line

```
router.get("*", serveStaticFiles("public/"));
```

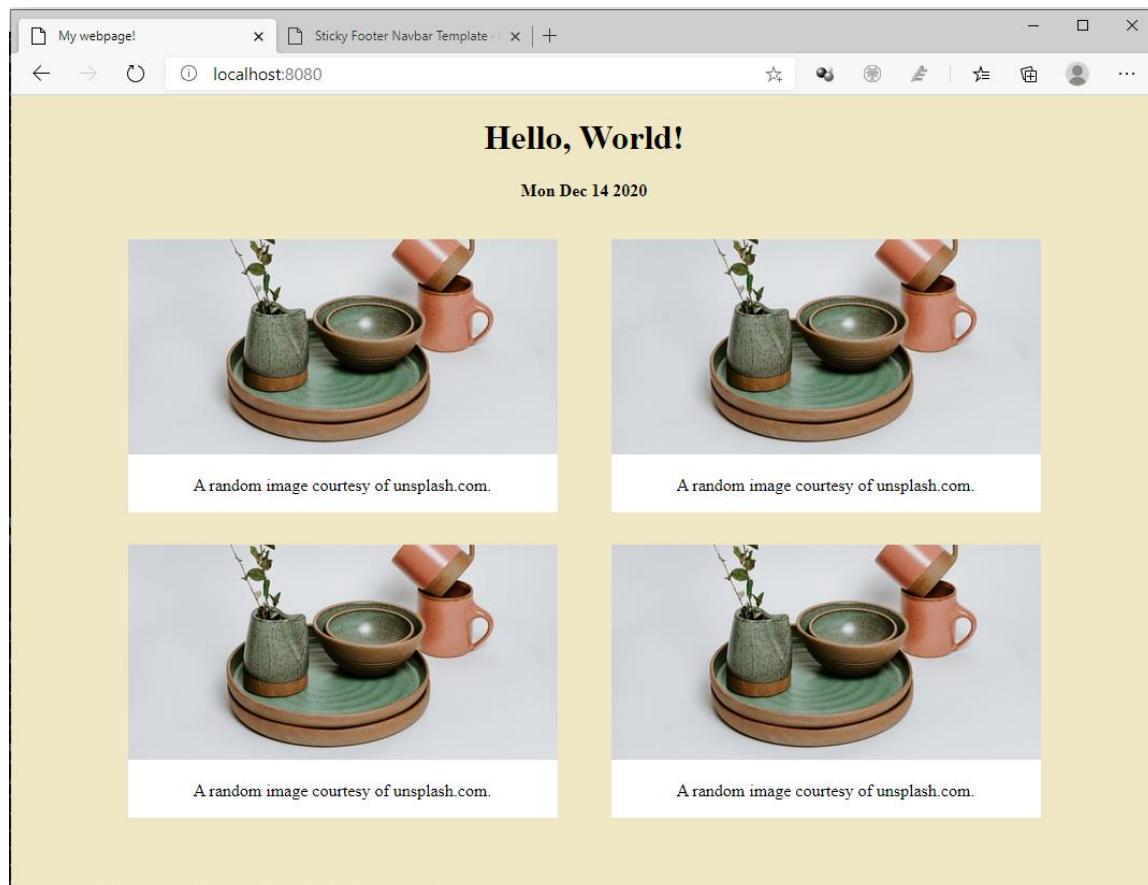
means that for all other files not indicated by any route, look for them inside the public/ folder.

We indicated in the helloagain.html page that the stylesheet and the javascript files are in /public/css/ and /public/js/ but without specifically naming the public/ folder.

```
<link rel="stylesheet" href="css/helloagain.css" />
<script async src="js/helloagain.js"></script>
```

We don't include the public/ folder in the path.

After refreshing the browser, something similar is what you will see:



The images you see will most likely be different as they are randomly selected every time you refresh the page.

Creating templates with Diet

Vibe.d comes with a templating engine called Diet.

Create views\simplehello.html with the following code

```
<!DOCTYPE html>
<html>
  <head>
    <title>Simple hello!</title>
  </head>
  <body>
    <h1>Simple hello, World!</h1>
    <h2>How are you doing today?</h2>
  </body>
</html>
```

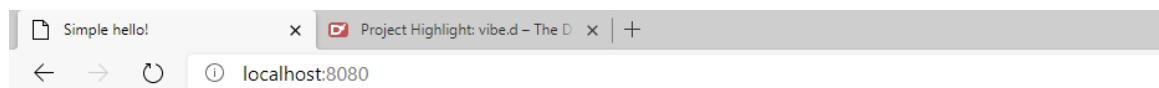
Edit the source\app.d file to make it display the views\simplehello.html file

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];
    listenHTTP(settings, staticTemplate! "simplehello.html");

    logInfo("Please open http://127.0.0.1:8080/ in your browser.");
    runApplication();
}
```

Then, in your terminal, build and run the project and refresh the browser.



Simple hello, World!

How are you doing today?

We just tested that the new HTML file is working. Now, let's convert the HTML file into a Diet template file.

First, save the views\simplehello.html file as views\simplehello.dt and edit views\simplehello.dt to make it look like this:

```
doctype html
html
  head
    title Simple hello using a template!
  body
    h1 Simple hello world from a template!
    h2 How are you doing today?
    | Please visit the home of <a href="https://vibed.org/">Vibe.d</a>
```

Then edit app.d to make it display simplehello.dt instead:

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];
    listenHTTP(settings, staticTemplate! "simplehello.dt");

    logInfo("Please open http://127.0.0.1:8080/ in your browser.");
    runApplication();
}
```

In your terminal, stop the app if it is still running (Ctrl-C), then compile and run.

```
dub --force
```

We sometimes use the --force option to tell dub to recompile everything again, just in case it fails to see that some files were changed and needed to be recompiled or that there are updated libraries that needed to be downloaded.

Refresh your browser to see the changes.



Simple hello world from a template!

How are you doing today?

Please visit the home of [Vibe.d](https://vibed.org/)

The Diet template filename extension is .dt. Diet template syntax is based on Jade templates (<http://jade-lang.com/>).

Here are some of the syntax rules:

The first word on a line is usually an HTML tag.

```
span Hello, world!
br
```

| (pipe) tag at the start of a line means the line is composed of plain text.

```
| This line is plain text but can include <a href="/">HTML</a>
```

:javascript tag at the start of a line means JavaScript code on the following indented lines.

```
:javascript
  function greet()
  {
    var greeting = "Hello, world!";
    alert(greeting);
  }
```

:css tag at the start of a line means CSS code on the following indented lines.

```
:css
  .alert-message
  {
    font-weight: bold;
    color: brown;
    text-align: center;
  }
```

The attributes of a tag are comma-separated values inside parentheses.

```
div(class="high-status", id="high-status-1");
```

A tag followed by a . (dot) followed by an identifier name means a CSS class name.

```
div.high-status
```

A tag followed by a # (hash) followed by an identifier name means a CSS ID name.

```
div#high-status-1
```

CSS class names and IDs can be strung together.

```
div.high-status#high-status-1
```

CSS classes can be strung together with dots.

```
div.high-status.row-major.row-inner
```

Plain text following a tag should be separated by a space.

```
div Hello world!      // note the space after div
div(class="high-status") Hello, world! // note the space after parenthesis
```

A tag ending in a . (dot) means multi-line plain text follows.

```
div.
  These lines are all simple text
  but may contain HTML code
  like <a href="/">this</a>.
```

Plain text may contain HTML code.

```
| Like <a href="/">this</a>.
```

Nesting of elements is indicated by adding indentation.

```
div Like this: this div encloses the tags below
  span so this span is inside the div
    div and this div is inside the span
      | and there is no need for a closing tag
```

A more interesting template page

Let's have a more interesting example.

Create another file named views\notsosimplehello.html with the following content

```
<!DOCTYPE html>
<html>
  <head>
    <title>Demo site</title>
  </head>
  <body>
    <header>
      <h2>Welcome to our not so simple site</h2>
    </header>
    <nav>
      <ul>
        <li><a href="about.html">About us</a></li>
        <li><a href="events.html">Events</a></li>
        <li><a href="contact.html">Contact us</a></li>
      </ul>
    </nav>
    <article>
      <h1>Welcome to our not so simple site!</h1>
      <div>
        Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.
      </div>
    </article>
    <footer>
      <p>Copyright 2021 Notsosimple Company</p>
    </footer>
  </body>
</html>
```

Feel free to write your own text here as the Lorem ipsum text is just filler.

Edit source\app.d to use the new file.

```
import vibe.vibe;
void main()
{
```

```

auto settings = new HTTPServerSettings();
settings.port = 8080;
settings.bindAddresses = [":1", "127.0.0.1"];
listenHTTP(settings, staticTemplate("notsosimplehello.html"));

logInfo("Please open http://127.0.0.1:8080/ in your browser.");
runApplication();
}

```

Then compile and run, then refresh your browser



Now that we have tested that the HTML page is working, let's convert it into a Diet template. Save notsosimplehello.html as notsosimplehello.dt and convert it to a template.

```

html
  head
    title Demo site
  body
    header
      h2 Welcome to our not so simple site
    nav
      ul
        li <a href="/about">About us</a>
        li <a href="/events">Events</a>
        li <a href="/contact">Contact us</a>
    article
      h1 Welcome to our not so simple site!
      div.
        Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book.

```

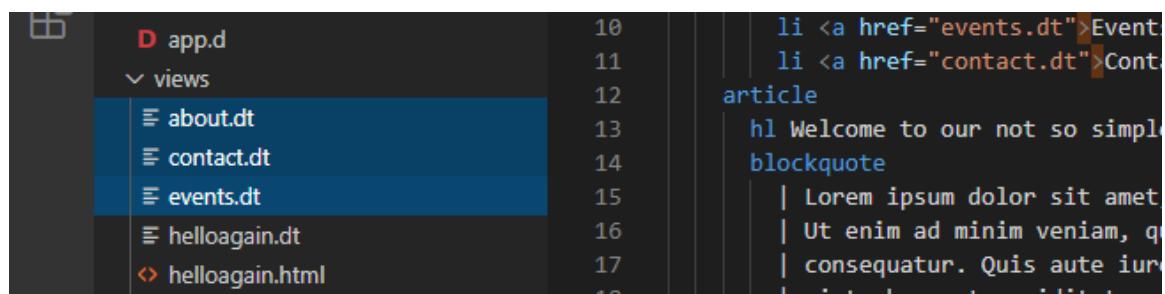
footer

```
p Copyright 2021 Notsosimple Company
```

Take note of the dot (.) after the div. It means all the indented lines that follow are plain text (which could include HTML).

It feels strange at first but you realize there is less to type when you use Diet templates.

Copy notsosimplehello.dt into about.dt, events.dt and contact.dt inside the views folder (or File->Save As... three times)



```
D app.d
  ↘ views
    ≡ about.dt
    ≡ contact.dt
    ≡ events.dt
    ≡ helloagain.dt
    ◊ helloagain.html
```

10	li Event
11	li Cont
12	article
13	h1 Welcome to our not so simpl
14	blockquote
15	Lorem ipsum dolor sit amet
16	Ut enim ad minim veniam, q
17	consequatur. Quis aute iur
18	et laboris nisi ut alit

Make minor changes to the new files so they each display a different message.

views\about.dt

```
html
  head
    title Demo site
  body
    header
      h2 This is the About Us page
    nav
      ul
        li <a href="/about">About us</a>
        li <a href="/events">Events</a>
        li <a href="/contact">Contact us</a>
    article
      div.
        Lorem Ipsum is simply dummy text of the printing and typesetting
        industry. Lorem Ipsum has been the industry's standard dummy
        text ever since the 1500s, when an unknown printer took a galley
        of type and scrambled it to make a type specimen book.
    footer
      p Copyright 2021 Notsosimple Company
```

views\contact.dt

```
html
  head
    title Demo site
  body
```

```

header
  h2 This is the Contact Us page
nav
  ul
    li <a href="/about">About us</a>
    li <a href="/events">Events</a>
    li <a href="/contact">Contact us</a>
article
  div.
    Lorem Ipsum is simply dummy text of the printing and typesetting
    industry. Lorem Ipsum has been the industry's standard dummy
    text ever since the 1500s, when an unknown printer took a galley
    of type and scrambled it to make a type specimen book.
footer
  p Copyright 2021 Notsosimple Company

```

views\events.dt

```

html
  head
    title Demo site
  body
    header
      h2 This is the Events page
    nav
      ul
        li <a href="/about">About us</a>
        li <a href="/events">Events</a>
        li <a href="/contact">Contact us</a>
    article
      div.
        Lorem Ipsum is simply dummy text of the printing and typesetting
        industry. Lorem Ipsum has been the industry's standard dummy
        text ever since the 1500s, when an unknown printer took a galley
        of type and scrambled it to make a type specimen book.
    footer
      p Copyright 2021 Notsosimple Company

```

Next, edit source\app.d to use views\notsosimplehello.dt instead

```

import vibe.vibe;

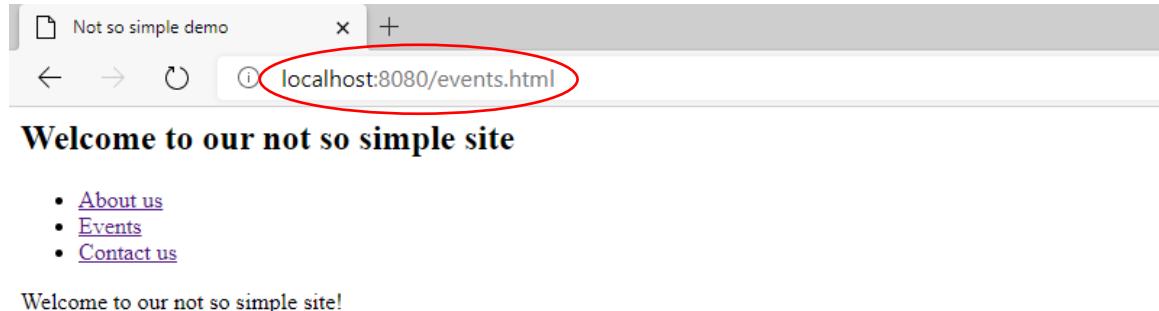
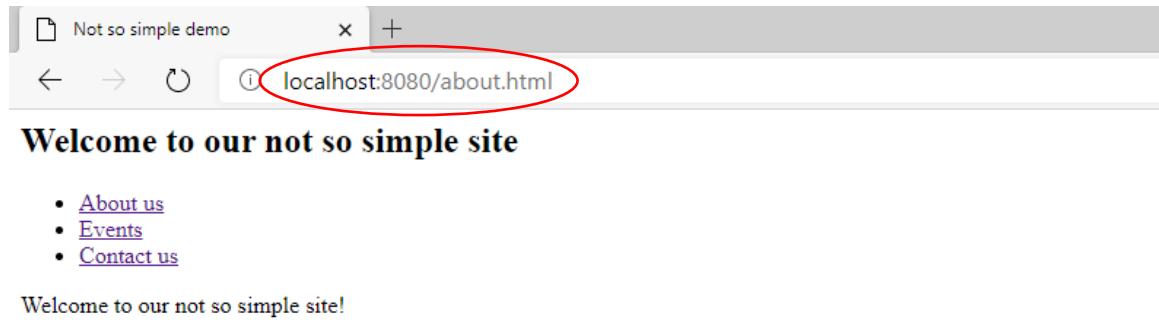
void main()
{
  auto settings = new HTTPServerSettings;
  settings.port = 8080;
  settings.bindAddresses = [":1", "127.0.0.1"];
  listenHTTP(settings, staticTemplate! "notsosimplehello.dt");

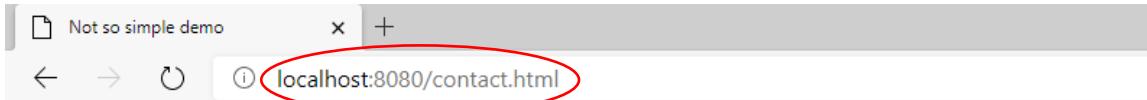
```

```
    logInfo("Please open http://127.0.0.1:8080/ in your browser.");
    runApplication();
}
```

Then compile, run and refresh your browser. If you see no changes, that means you did good.

But when you click on the links, it is still showing the same page although the URL on the browser address bar changes





Welcome to our not so simple site

- [About us](#)
- [Events](#)
- [Contact us](#)

Welcome to our not so simple site!

That's because the server thinks we are just going to display the same page, notsosimplehello.dt, no matter the URL.

We need a *router* to tell the server to display a certain page when given a certain URL. This matching of URLs to pages or actions is called *routing*.

Edit source\app.d to make use of a router to display the appropriate page when given a specified URL.

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    auto router = new URLRouter;
    router.get("/", staticTemplate! "notsosimplehello.dt");
    router.get("/about", staticTemplate! "about.dt");
    router.get("/events", staticTemplate! "events.dt");
    router.get("/contact", staticTemplate! "contact.dt");

    listenHTTP(settings, router);
    runApplication();
}
```

In D, we can make use of the `auto` keyword if the data type can be deduced. The line

```
auto router = new URLRouter;
```

means `router` is a `URLRouter` class type. We could have also written this as

```
URLRouter router = new URLRouter();
```

But auto is a lot shorter to write than URLRouter, so we use it. Also, if there are no arguments, the parentheses are optional, so URLRouter() is the same as URLRouter.

The line

```
router.get("/", staticTemplate! "notsosimplehello.dt");
```

tells us that when the browser requests for the URL

<http://localhost:8080/>

the server should display the file notsosimplehello.dt.

The line

```
router.get("/about", staticTemplate! "about.dt");
```

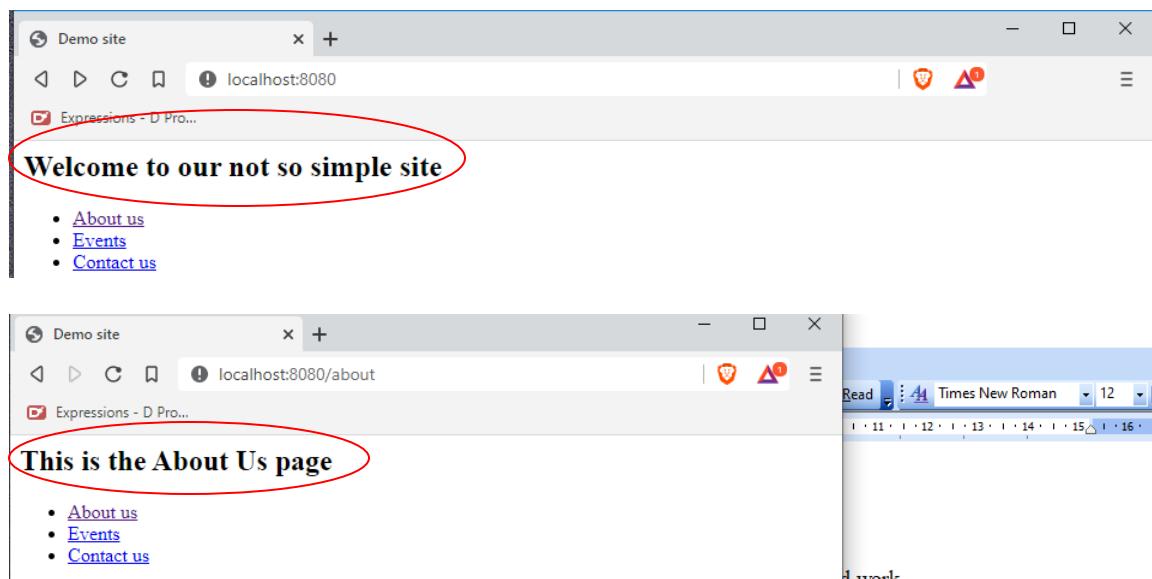
means that when the browser asks for

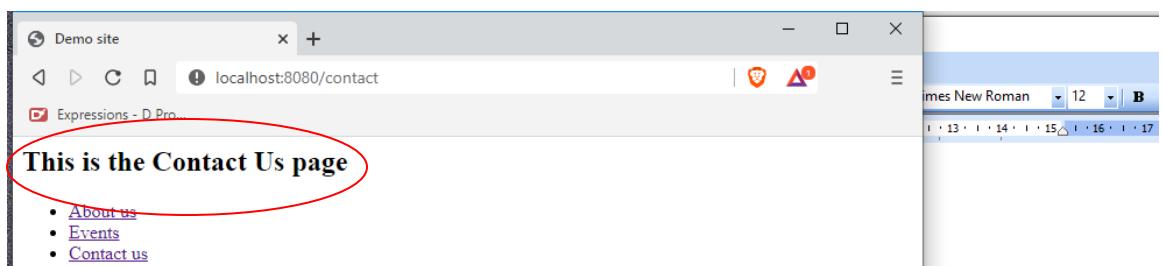
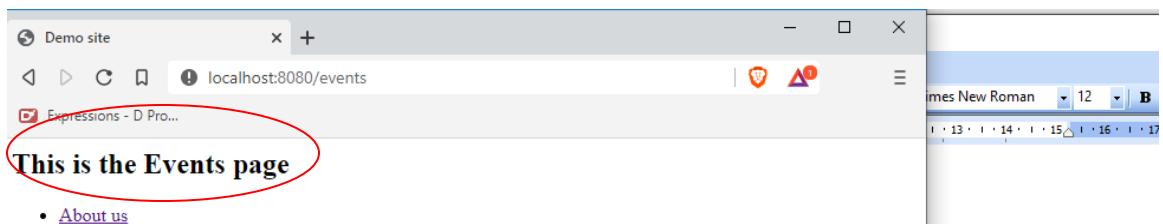
<http://localhost:8080/about>

then about.dt should be rendered.

And so on.

Compile, run and refresh your browser. Now the links should work.





Making use of your own functions

You can also make calls to your own functions. Add the following link to notsosimplehello.dt.

```
html
  head
    title Demo site
  body
    header
      h2 Welcome to our not so simple site
    nav
      ul
        li <a href="/about">About us</a>
        li <a href="/events">Events</a>
        li <a href="/contact">Contact us</a>
        li <a href="/helloagain">Hello again!</a>
    article
      h1 Welcome to our not so simple site!
      div Lorem Ipsum is simply dummy text of the printing industry.
    footer
      p Copyright 2021 Notsosimple Company
```

And edit source\app.d to call your own user-created function helloAgain()

```
import vibe.vibe;

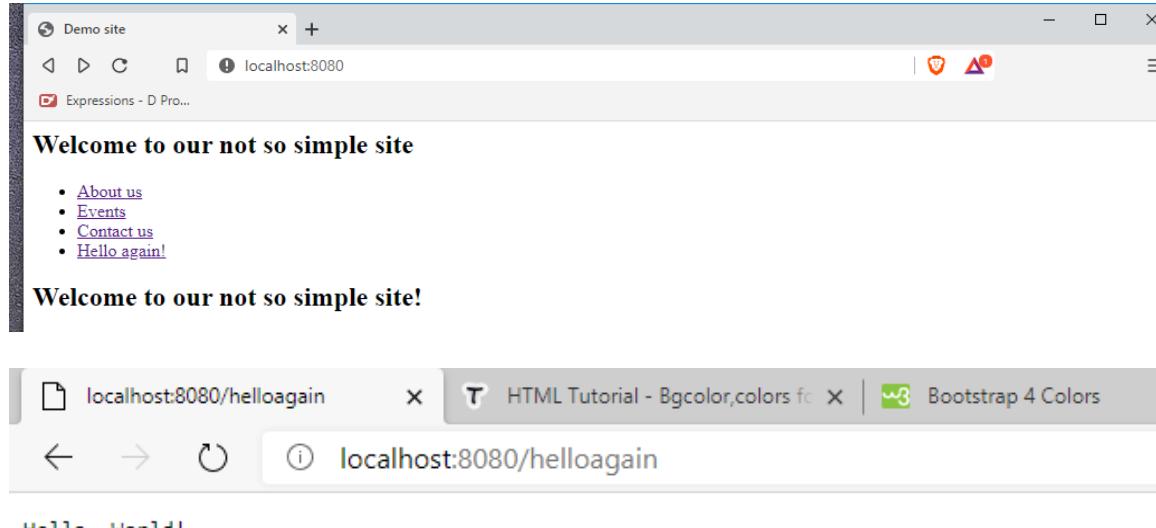
void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    auto router = new URLRouter;
    router.get("/", staticTemplate! "notsosimplehello.dt");
    router.get("/about", staticTemplate! "about.dt");
    router.get("/events", staticTemplate! "events.dt");
    router.get("/contact", staticTemplate! "contact.dt");
    router.get("/helloagain", &helloAgain);

    listenHTTP(settings, router);
    runApplication();
}
```

```
void helloAgain(HTTPRequest req, HTTPResponse res)
{
    string title = "Hello, World!";
    res.writeBody(title);
}
```

Compile, run, refresh your browser, then click on the Hello again! link and you will see that the new function gets called.



The helloAgain() function gets called.

Using templates to reduce code duplication

In the last example, there were some code duplication. Each and every page has the same navigation links as well as header and footer. If we want to edit the links, for example, we will have to go through each and every file and edit them. Although there is that super wonderful tool called copy-and-paste, doing manual changes to different files is still fraught with errors, especially when the files we are editing are getting bigger and bigger.

To reduce code duplication and errors in maintenance, Diet templates provide a mechanism using inheritance.

Let us create a new project named helloblocks.

```
C:\vibeprojects\hello>cd ..
```

```
C:\vibeprojects>dub init helloblocks -t vibe.d
```

```
Package recipe format (sdl/json) [json]:
```

```
Name [helloblocks]:
```

```
Description [A simple vibe.d server application.]:
```

```
Author name [rey]:
```

```
License [proprietary]:
```

```
Copyright string [Copyright © 2021, rey]:
```

```
Add dependency (leave empty to skip) []:
```

```
Successfully created an empty project in 'C:\vibeprojects\helloblocks'.
```

```
Package successfully created in helloblocks
```

```
C:\vibeprojects>cd helloblocks
```

```
C:\vibeprojects\helloblocks>
```

Create views\mainlayout.dt.

```
html
  head
    title Using templates
  body
    nav
      ul
        li <a href="/">Home</a>
        li <a href="/about">About us</a>
        li <a href="/events">Events</a>
        li <a href="/contact">Contact us</a>
        li <a href="/helloagain">Hello again!</a>
    block content
  footer
```

```
p Copyright 2021 Notsosimple Company
```

The line

```
block content
```

means we have named that part, or ‘block’, of the layout as ‘content’. You can give the block any name as long as you give the same name to the other templates that inherit from this template. There is nothing inside that block because it will be defined by the other pages.

Create views\home.dt.

```
extends mainlayout
block content
  article
    h1 Welcome to our not so simple site!
```

The first line indicates that this template inherits from mainlayout.dt, meaning all of mainlayout.dt are included in this template.

The second line, block content, indicates that the following lines will appear in the corresponding content block indicated by mainlayout.dt.

Create views\about.dt.

```
extends mainlayout
block content
  article
    h1 About us
```

Create views\events.dt.

```
extends mainlayout
block content
  article
    h1 Upcoming events
```

Create views\contact.dt.

```
extends mainlayout
block content
  article
    h1 Contact us
```

Next, edit source\app.d to make it look like this:

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

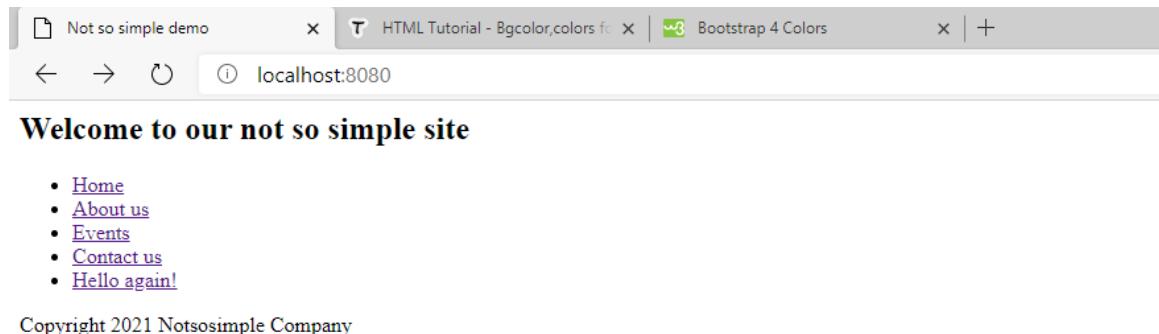
    auto router = new URLRouter;
    router.get("/", staticTemplate!"home.dt");
    router.get("/about", staticTemplate!"about.dt");
    router.get("/events", staticTemplate!"events.dt");
    router.get("/contact", staticTemplate!"contact.dt");
    router.get("/helloagain", &helloAgain);
    router.get("*", serveStaticFiles("public/"));

    listenHTTP(settings, router);
    runApplication();
}

void helloAgain(HTTPServerRequest req, HTTPServerResponse res)
{
    string title = "Hello, World!";
    res.writeBody(title);
}
```

By creating a layout template and using the block mechanism, we placed the repeating tags in one template while leaving the other templates with only their specific contents. We named the part that changes as the content block and we indicated in the other pages that their contents will appear in that content block of the layout.

Compile and run the project with dub, then refresh the browser.



Using include in templates

There is also an include directive in Diet templates that come in handy when breaking up templates into smaller parts.

Create views\navbar.dt and extract from views\mainlayout.dt this content:

```
nav
  ul
    li <a href="/">Home</a>
    li <a href="/about">About us</a>
    li <a href="/events">Events</a>
    li <a href="/contact">Contact us</a>
    li <a href="/helloagain">Hello again!</a>
```

Then create views\footer.dt.

```
footer
  p Copyright 2021 Notsosimple Company
```

Then edit views\mainlayout.dt.

```
html
  head
    title Using templates
  body
    include navbar
    block content
    include footer
```

Try to compile and run the project.

```
C:\vibeprojects\helloblocks>dub
```

Sometimes, you get errors like this.

Compiling Diet HTML template home.dt...

```
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\defs.d(34,3): Error: "mainlayout.dt(5): Mismatched indentation style."
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(952,11):           called from here: enforce(is_text_line ||
indent.length == 0LU, delegate string() pure nothrow @nogc @safe => "Mismatched
indentation style.", loc)
```

C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,76): called from here: parseDietRaw(f)
C:\D\dm2\windows\bin\..\..\src\phobos\std\algorithm\iteration.d(625,19): called
from here: __lambda2(front(this._input))
C:\D\dm2\windows\bin\..\..\src\phobos\std\array.d(112,9): called from here:
_r2036.front()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,81): called from here: array(map(files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,90): called from here:
parseDiet(cast(const(InputFile)[])_diet_files)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,61): called from here:
applyTraits(parseDiet(cast(const(InputFile)[])_diet_files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,47): called from here: _diet_nodes()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,35): called from here: getHTMLMixin(_diet_nodes(),
"_diet_output", HTMLOutputStyle.pretty)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(194,22): Error: template instance
diet.html.realCompileHTMLDietFileString!("home.dt", contents, DefaultDietFilters) error
instantiating
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(40,30): instantiated from here:
compileHTMLDietFileString!("home.dt", contents, req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(351,2): instantiated from here:
compileHTMLDietFile!("home.dt", req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(276,6): instantiated from here: render!("home.dt", req)
source\app.d(10,19): instantiated from here: staticTemplate!"home.dt"
Compiling Diet HTML template about.dt...
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\defs.d(34,3): Error: "mainlayout.dt(5): Mismatched indentation style."
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(952,11): called from here: enforce(is_text_line ||
indent.length == OLU, delegate string() pure nothrow @nogc @safe => "Mismatched
indentation style.", loc)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,76): called from here: parseDietRaw(f)
C:\D\dm2\windows\bin\..\..\src\phobos\std\algorithm\iteration.d(625,19): called
from here: __lambda2(front(this._input))

C:\D\dm2\windows\bin\..\..\src\phobos\std\array.d(112,9): called from here:
__r2036.front()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,81): called from here: array(map(files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,90): called from here:
parseDiet(cast(const(InputFile)[])_diet_files)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,61): called from here:
applyTraits(parseDiet(cast(const(InputFile)[])_diet_files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,47): called from here: _diet_nodes()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,35): called from here: getHTMLMixin(_diet_nodes(),
"_diet_output", HTMLOutputStyle.pretty)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(194,22): Error: template instance
diet.html.realCompileHTMLDietFileString!("about.dt", contents, DefaultDietFilters) error
instantiating
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(40,30): instantiated from here:
compileHTMLDietFileString!("about.dt", contents, req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(351,2): instantiated from here:
compileHTMLDietFile!("about.dt", req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(276,6): instantiated from here: render!("about.dt", req)
source\app.d(11,24): instantiated from here: staticTemplate!"about.dt"
Compiling Diet HTML template events.dt...
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\defs.d(34,3): Error: "mainlayout.dt(5): Mismatched indentation style."
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(952,11): called from here: enforcep(is_text_line ||
indent.length == OLU, delegate string() pure nothrow @nogc @safe => "Mismatched
indentation style.", loc)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,76): called from here: parseDietRaw(f)
C:\D\dm2\windows\bin\..\..\src\phobos\std\algorithm\iteration.d(625,19): called
from here: __lambda2(front(this._input))
C:\D\dm2\windows\bin\..\..\src\phobos\std\array.d(112,9): called from here:
__r2036.front()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,81): called from here: array(map(files))

```
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,90):                                called      from      here:
parseDiet(cast(const(InputFile)[])_diet_files)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,61):                                called      from      here:
applyTraits(parseDiet(cast(const(InputFile)[])_diet_files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,47):    called from here: _diet_nodes()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,35):    called from here: getHTMLMixin(_diet_nodes(),
"_diet_output", HTMLOutputStyle.pretty)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(194,22):    Error:          template           instance
diet.html.realCompileHTMLDietFileString!("events.dt", contents, DefaultDietFilters) error
instantiating
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(40,30):                                instantiated   from      here:
compileHTMLDietFileString!("events.dt", contents, req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(351,2):                                instantiated   from      here:
compileHTMLDietFile!("events.dt", req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(276,6):    instantiated from here: render!("events.dt", req)
source\app.d(12,25):    instantiated from here: staticTemplate!"events.dt"
Compiling Diet HTML template contact.dt...
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\defs.d(34,3): Error: "mainlayout.dt(5): Mismatched indentation style."
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(952,11):    called from here: enforce(is_text_line || 
indent.length == 0LU, delegate string() pure nothrow @nogc @safe => "Mismatched
indentation style.", loc)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,76):    called from here: parseDietRaw(f)
C:\D\dm2\windows\bin\..\..\src\phobos\std\algorithm\iteration.d(625,19):    called
from here: __lambda2(front(this._input))
C:\D\dm2\windows\bin\..\..\src\phobos\std\array.d(112,9):    called from here:
__r2036.front()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\parser.d(74,81):    called from here: array(map(files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,90):                                called      from      here:
parseDiet(cast(const(InputFile)[])_diet_files)
```

```

C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(150,61):                                called      from      here:
applyTraits(parseDiet(cast(const(InputFile)[])_diet_files))
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,47):      called from here: _diet_nodes()
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(157,35):      called from here: getHTMLMixin(_diet_nodes()),
"_diet_output", HTMLOutputStyle.pretty)
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(194,22):          Error:          template          instance
diet.html.realCompileHTMLDietFileString!("contact.dt",   contents,   DefaultDietFilters)
error instantiating
C:\Users\rey\AppData\Local\dub\packages\diet-ng-1.7.4\diet-
ng\source\diet\html.d(40,30):          instantiated      from      here:
compileHTMLDietFileString!("contact.dt", contents, req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(351,2):          instantiated      from      here:
compileHTMLDietFile!("contact.dt", req, DefaultDietFilters)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(276,6):      instantiated from here: render!("contact.dt", req)
source\app.d(13,26):      instantiated from here: staticTemplate!"contact.dt"
C:\D\dm2\windows\bin\dm.exe failed with exit code 1.

```

Sometimes all you need to do is erase all the indentations and manually indent them again using tabs. After editing, compile, run and refresh the browser.

This should be the output.



Using templates

localhost:8080/about

Expressions - D Pro...

- [Home](#)
- [About us](#)
- [Events](#)
- [Contact us](#)
- [Hello again!](#)

About us

Copyright 2021 Notsosimple Company

Using templates

localhost:8080/events

Expressions - D Pro...

- [Home](#)
- [About us](#)
- [Events](#)
- [Contact us](#)
- [Hello again!](#)

Upcoming events

Copyright 2021 Notsosimple Company

Using templates

localhost:8080/contact

Expressions - D Pro...

- [Home](#)
- [About us](#)
- [Events](#)
- [Contact us](#)
- [Hello again!](#)

Contact us

Copyright 2021 Notsosimple Company

localhost:8080/helloagain

localhost:8080/helloagain

Expressions - D Pro...

Hello, World!

A Bootstrap template with a sticky navbar and a sticky footer

Bootstrap is the most popular CSS/JavaScript designing tool, so let's follow the crowd.

A sticky navbar means a navigation bar that always stay on top of the page, no matter the length and shape of the web page. A sticky footer always stay at the bottom.

Stop the running app with Ctrl-C, then create a new project named mybootstrap.

```
c:\vibeprojects\hello>cd ..
```

```
c:\vibeprojects>dub init mybootstrap -t vibe.d
```

```
Package recipe format (sdl/json) [json]:
```

```
Name [mybootstrap]:
```

```
Description [A simple vibe.d server application.]:
```

```
Author name [lenovo pc]:
```

```
License [proprietary]:
```

```
Copyright string [Copyright © 2020, lenovo pc]:
```

```
Add dependency (leave empty to skip) []:
```

```
Successfully created an empty project in 'c:\vibeprojects\mybootstrap'.
```

```
Package successfully created in mybootstrap
```

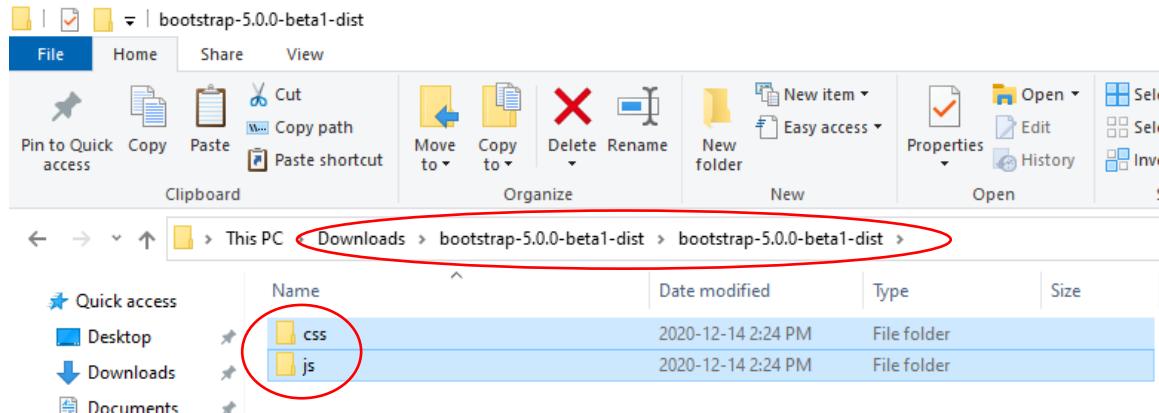
```
c:\vibeprojects>cd mybootstrap
```

```
c:\vibeprojects\mybootstrap>
```

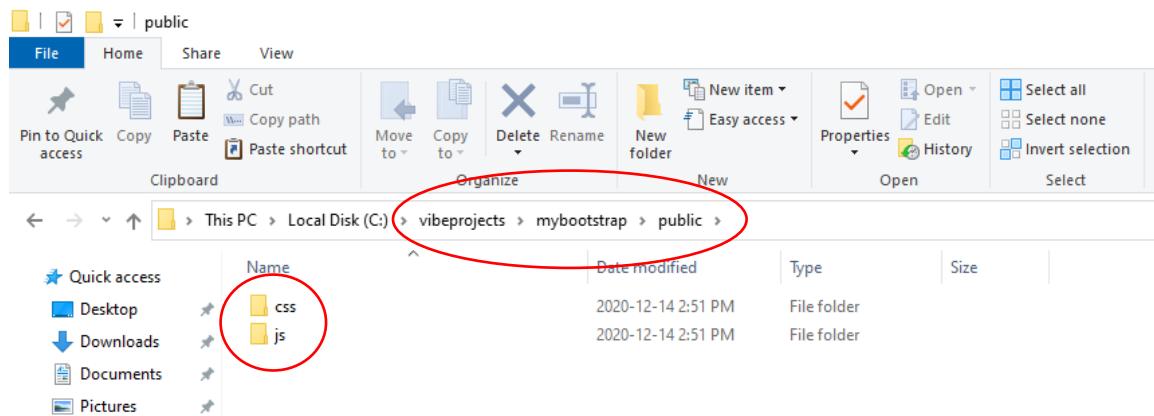
To use Bootstrap in our web page, we should have access to the Bootstrap files. Download the Bootstrap files here: <https://getbootstrap.com/docs/5.0/getting-started/download/>

The screenshot shows a browser window with the URL <https://getbootstrap.com/docs/5.0/getting-started/download/>. The page title is "Compiled CSS and JS". On the left, there's a sidebar with a "Getting started" section containing links like "Introduction", "Download", "Contents", "Browsers & devices", "JavaScript", "Build tools", "Webpack", "Accessibility", and three dots. The main content area starts with a paragraph about downloading compiled code for Bootstrap v5.0.0-beta1, which includes compiled and minified CSS bundles and JavaScript plugins. Below this is a note stating that the download doesn't include documentation, source files, or optional dependencies like Popper. At the bottom of this section is a prominent blue "Download" button, which is circled in red.

After extracting, copy the whole css and the js folders and paste them into the public folder of your project



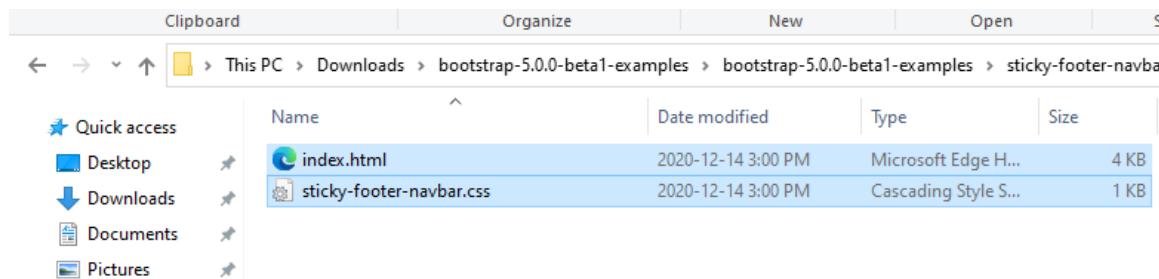
Copy those files into this folder



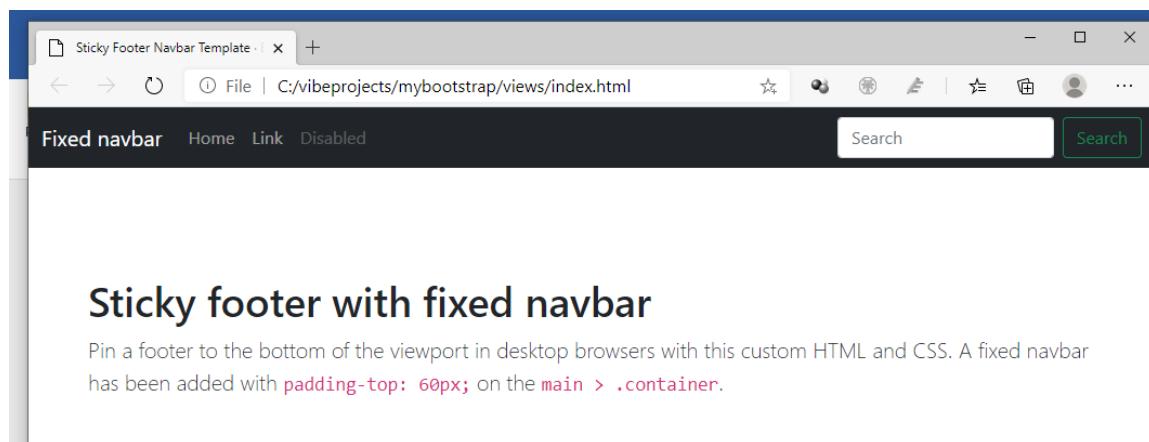
On the same Bootstrap page, download and extract the Bootstrap examples

The screenshot shows the Bootstrap website's 'Getting started' section. It includes links for Introduction, Download, Contents, Browsers & devices, JavaScript, Build tools, Webpack, Accessibility, RFS, and RTL. To the right, there is information about build tools and a 'Download source' button. Below this is the 'Examples' section, which contains a link to download examples and a 'Download Examples' button. A red circle highlights the 'Download Examples' button. At the bottom, there is a 'CDN via jsDelivr' link.

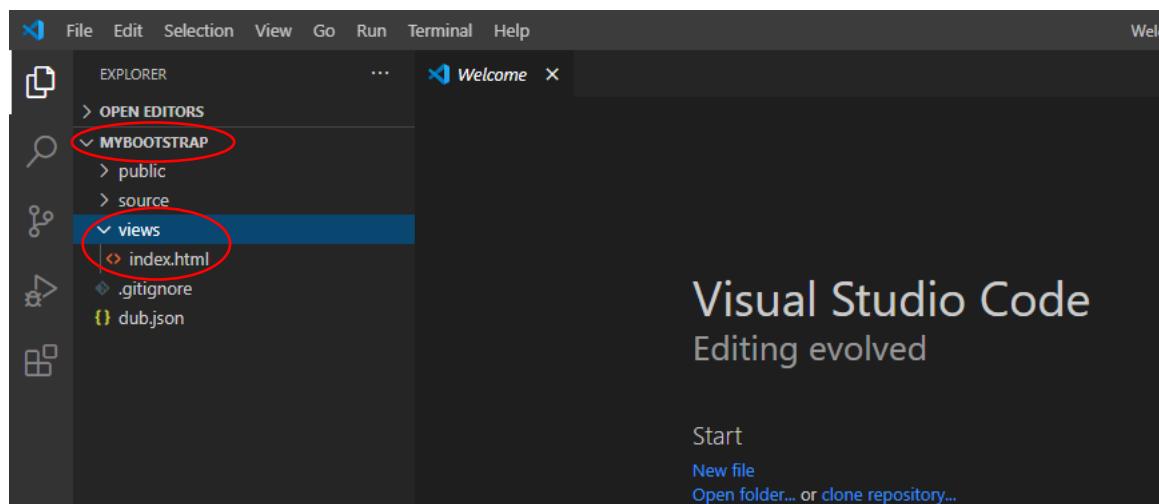
After extraction, you will find there are two files inside the sticky-footer-navbar folder



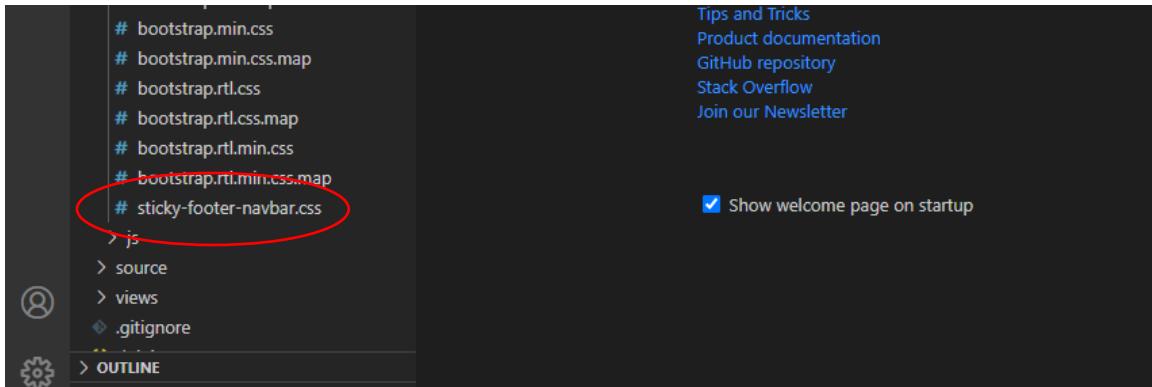
To test and see how it looks on the browser, simply double-click on index.html!



Copy the index.html file into the views\ folder of your project. Inside VS Code, open the c:\vibeprojects\mybootstrap folder.



And copy the sticky-footer-navbar\sticky-footer-navbar.css into public\css\sticky-footer-navbar.css of the mybootstrap project.



Edit `views\index.html` and change the URLs of the CSS and the JavaScript files

```
<!doctype html>
<html lang="en" class="h-100">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta name="description" content="">
    <meta name="author" content="Mark Otto, Jacob Thornton, and Bootstrap contributors">
    <meta name="generator" content="Hugo 0.79.0">
    <title>Sticky Footer Navbar Template · Bootstrap v5.0</title>
    <link rel="canonical" href="https://getbootstrap.com/docs/5.0/examples/sticky-footer-navbar/">
    <link href="css/bootstrap.min.css" rel="stylesheet">
  :css
    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }
    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
      }
    }
    <link href="css/sticky-footer-navbar.css" rel="stylesheet">
  </head>
  <body class="d-flex flex-column h-100">
    <header>
      <!-- fixed navbar -->
      <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
```

```

<div class="container-fluid">
    <a class="navbar-brand" href="#">Fixed navbar</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-
bs-target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-
label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarCollapse">
        <ul class="navbar-nav me-auto mb-2 mb-md-0">
            <li class="nav-item active">
                <a class="nav-link" aria-current="page" href="#">Home</a>
            </li>
            <li class="nav-item">
                <a class="nav-link" href="#">Link</a>
            </li>
            <li class="nav-item">
                <a class="nav-link disabled" href="#" tabindex="-1" aria-
disabled="true">Disabled</a>
            </li>
        </ul>
    </div>
</div>
</nav>
</header>
<!-- Begin page content -->
<main class="flex-shrink-0">
    <div class="container">
        <h1 class="mt-5">Sticky footer with fixed navbar</h1>
        <p class="lead">Pin a footer to the bottom of the viewport in desktop browsers
with this custom HTML and CSS. A fixed navbar has been added with <code class="small">p
adding-top: 60px;</code> on the <code class="small">main &gt; .container</code>. </p>
    </div>
</main>
<!-- the footer -->
<footer class="footer mt-auto py-3 bg-light">
    <div class="container">
        <span class="text-muted">Place sticky footer content here.</span>
    </div>
</footer>
<script src="js/bootstrap.bundle.min.js"></script>
</body>
</html>

```

Edit source\app.d.

```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    auto router = new URLRouter;
    router.get("/", staticTemplate! "index.html");
    router.get("*", serveStaticFiles("public/"));

    listenHTTP(settings, router);

    runApplication();
}
```

The `serveStaticFiles("public/")` function indicates that for other files (or URLs) not specifically mentioned by the router, look for them under the `public/` folder. This way, the line

```
<link href="css/bootstrap.min.css" rel="stylesheet">
```

will be expanded to

```
<link href='public/css/bootstrap.min.css' rel="stylesheet">
```

and the line

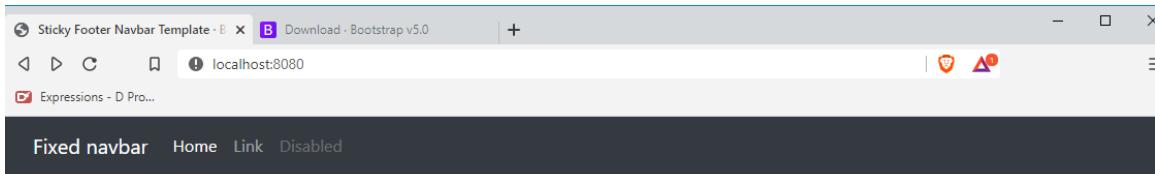
```
<script src="js/bootstrap.bundle.min.js"></script>
```

will be expanded to

```
<script src='public/js/bootstrap.bundle.min.js'></script>
```

We do not include the `public/` part in the path because the server already knows.

Compile, run and refresh the browser.



Sticky footer with fixed navbar

Pin a footer to the bottom of the viewport in desktop browsers with this custom HTML and CSS. A fixed navbar has been added with `padding-top: 60px;` on the `main > .container`.

We have confirmed that the page is working. We see the navigation bar at the top and the footer at the bottom.

Now let's convert index.html into a template file.

Converting index.html into a Diet template

First, let's convert the whole page as one template. Let's transform index.html into a Diet template.

Create views\layout.dt.

```
html(lang="en", class="h-100")
  head
    meta(charset="utf-8")
    meta(name="viewport", content="width=device-width, initial-scale=1")
    meta(name="description", content="")
    meta(name="author", content="Mark Otto, Jacob Thornton and others")
    meta(name="generator", content="Hugo 0.79.0")
    title Sticky Footer Navbar Template • Bootstrap v5.0
    link(href="css/bootstrap.min.css", rel="stylesheet")
  :css
    .bd-placeholder-img {
      font-size: 1.125rem;
      text-anchor: middle;
      -webkit-user-select: none;
      -moz-user-select: none;
      user-select: none;
    }
    @media (min-width: 768px) {
      .bd-placeholder-img-lg {
        font-size: 3.5rem;
      }
    }
    link(href="css/sticky-footer-navbar.css", rel="stylesheet")
  body.d-flex.flex-column.h-100
    header
      //-- fixed navbar
      nav.navbar.navbar-expand-md.navbar-dark.fixed-top.bg-dark
        div.container-fluid
          a.navbar-brand(href="#") Fixed navbar
          button.navbar-toggler(type="button", data-bs-toggle="collapse", data-bs-target="#navbarCollapse", aria-controls="navbarCollapse", aria-expanded="false", aria-label="Toggle navigation")
            span.navbar-toggler-icon
          div.collapse.navbar-collapse#navbarCollapse
            ul.navbar-nav.me-auto.mb-2.mb-md-0
              li.nav-item
                a.nav-link.active(aria-current="page", href="#") Home
              li.nav-item
                a.nav-link(href="#") Link
              li.nav-item
```

```
a.nav-link.disabled(href="#", tabindex="-1", aria-
disabled="true") Disabled
// - Begin page content
main.flex-shrink-0
div.container
h1.mt-5 Sticky footer with fixed navbar
p.lead.
    Pin a footer to the bottom of the viewport in desktop browsers with this
    custom HTML and CSS. A fixed navbar has been added
    with <code class="small">padding-
top: 60px;</code> on the <code class="small">main &gt; .container</code>.
// - the footer -->
footer.footer.mt-auto.py-3.bg-light
div.container
span.text-muted Place sticky footer content here.
script(src="js/bootstrap.bundle.min.js")
```

There are some new things here. The line

```
<html lang="en" class="h-100">
```

has become

```
html(lang="en", class="h-100")
```

which could also be written as

```
html.h-100(lang="en")
```

and it will still be the same.

The line

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

has become

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

Since there are two attributes to the viewport meta tag, namely name and content, we have to separate them with a comma.

The line

```
<div class="collapse navbar-collapse" id="navbarCollapse">
```

has become

```
div/.collapse.navbar-collapse#navbarCollapse
```

Since collapse and navbar-collapse are CSS classes applied to the same <div> tag, we string them together with dots (.), and since navbarCollapse is a CSS ID, we appended it with a #.

Edit source\app.d to make it render layout.dt instead.

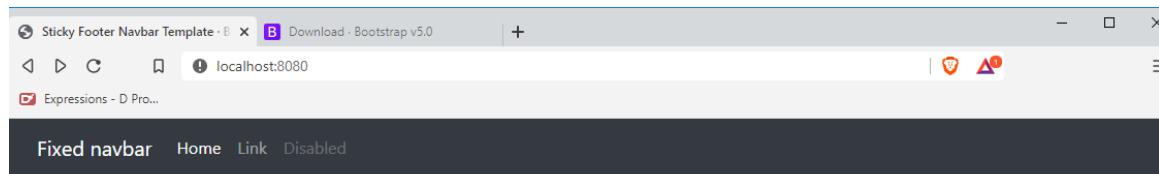
```
import vibe.vibe;

void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    auto router = new URLRouter;
    router.get("/", staticTemplate!("layout.dt"));
    router.get("*", serveStaticFiles("public/"));

    listenHTTP(settings, router);
    runApplication();
}
```

Compile, run and refresh your browser. If you find no changes, you did good.



Sticky footer with fixed navbar

Pin a footer to the bottom of the viewport in desktop browsers with this custom HTML and CSS. A fixed navbar has been added with `padding-top: 60px;` on the `main > .container`.

Now, time to break up the layout into smaller, manageable and logical chunks for ease of update.

Breaking up layout.dt into smaller parts with include

Let's break up views\layout.dt into manageable chunks to make the app easier to edit.

We will extract from views\layout.dt and paste them into different files.

Create views\headmeta.dt and extract from views\layout.dt these contents:

```
meta(charset="utf-8")
meta(name="viewport", content="width=device-width, initial-scale=1")
meta(name="description", content="")
meta(name="author", content="Mark Otto, Jacob Thornton, and Bootstrap contributors")
meta(name="generator", content="Hugo 0.79.0")
title Sticky Footer Navbar Template · Bootstrap v5.0
link(href="css/bootstrap.min.css", rel="stylesheet")
:css
.bd-placeholder-img {
  font-size: 1.125rem;
  text-anchor: middle;
  -webkit-user-select: none;
  -moz-user-select: none;
  user-select: none;
}
@media (min-width: 768px) {
  .bd-placeholder-img-lg {
    font-size: 3.5rem;
  }
}
link(href="css/sticky-footer-navbar.css", rel="stylesheet")
```

Create views\navbar.dt and extract from views\layout.dt these contents:

```
nav.navbar.navbar-expand-md.navbar-dark.fixed-top.bg-dark
  div.container-fluid
    a.navbar-brand(href="#") Fixed navbar
    button.navbar-toggler(type="button", data-bs-toggle="collapse", data-bs-target="#navbarCollapse", aria-controls="navbarCollapse", aria-expanded="false", aria-label="Toggle navigation")
      span.navbar-toggler-icon
    div.collapse.navbar-collapse#navbarCollapse
      ul.navbar-nav.me-auto.mb-2.mb-md-0
        li.nav-item
          a.nav-link.active(aria-current="page", href="#") Home
        li.nav-item
          a.nav-link(href="#") Link
```

```
li.nav-item
    a.nav-link.disabled(href="#", tabindex="-1", aria-
disabled="true") Disabled
```

Create views\footer.dt and extract from views\layout.dt these contents:

```
div.container
    span.text-muted Place sticky footer content here.
```

Create views\bootstrapscript.dt and extract from views\layout.dt these contents:

```
script(src="js/bootstrap.bundle.min.js")
```

Create views\home.dt (take note of the ending . (dot) in p.lead.)

```
extends layout.dt
block maincontent
    div.container
        h1.mt-5 Sticky footer with fixed navbar
        p.lead.
            Pin a footer to the bottom of the viewport in desktop browsers
            with this custom HTML and CSS. A fixed navbar has been added
            with <code class="small">padding-top: 60px;</code> on the
            <code class="small">main &gt; .container</code>.
```

The dot at the end of the p.lead. tag indicates that the next indented lines are plain text.

So this is what is left with layout.dt:

```
html(lang="en", class="h-100")
    head
        include headmeta.dt
    body.d-flex.flex-column.h-100
        header
            include navbar.dt
        main.flex-shrink-0
            block maincontent
        footer.footer.mt-auto.py-3.bg-light
            include footer.dt
            include bootstrapscript.dt
```

This time, we named the block with changing content dynamic. You can give the block any non-keyword name, just like the rules on variable names.

Edit source\app.d to make it show home.dt instead:

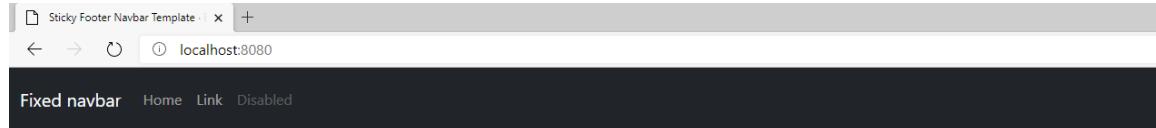
```
import vibe.vibe;
```

```
void main()
{
    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    auto router = new URLRouter;
    router.get("/", staticTemplate("home.dt"));
    router.get("*", serveStaticFiles("public/"));

    listenHTTP(settings, router);
    runApplication();
}
```

Compile, run and refresh the browser. If it showed the navbar and the footer as it did before, you did things right.



Sticky footer with fixed navbar

Pin a footer to the bottom of the viewport in desktop browsers with this custom HTML and CSS. A fixed navbar has been added to the top of the page.

Layout without Bootstrap: using CSS Grid

How about layout without Bootstrap?

Thankfully, standard CSS has CSS Grid. CSS Grid was designed to make it easier to lay out components on a page using standard CSS. It divides the page into a grid composed of rows and columns, like a table, which is similar to Bootstrap's row- and col- classes.

Let us try using an example.

Create views\cssgrid.html:

```
<html>
  <head>
    <title>CSS Grid Sample</title>
    <style>
      .container
      {
        margin: 0 auto;
        width: 100%;
        height: 100%;
        display: grid;
        grid-template-columns: 150px 150px 150px;
        grid-template-rows: 200px 200px 200px;
      }
      .entry
      {
        color: white;
        text-align: center;
        display: table-cell;
        vertical-align: middle;
      }
      #first { background-color: red; }
      #second { background-color: green; }
      #third { background-color: blue; }
      #fourth { background-color: teal; }
      #fifth { background-color: grey; }
      #sixth { background-color: brown; }
      #seventh { background-color: silver; }
      #eighth { background-color: maroon; }
      #ninth { background-color: cyan; }
    </style>
  </head>
  <body>
    <div class="container">
```

```
<div class="entry" id="first">1st</div>
<div class="entry" id="second">2nd</div>
<div class="entry" id="third">3rd</div>
<div class="entry" id="fourth">4th</div>
<div class="entry" id="fifth">5th</div>
<div class="entry" id="sixth">6th</div>
<div class="entry" id="seventh">7th</div>
<div class="entry" id="eighth">8th</div>
<div class="entry" id="ninth">9th</div>
</div>
</body>
</html>
```

The line

```
grid-template-columns: 150px 150px 150px;
```

means we are defining three columns, with each column 150 pixels wide.

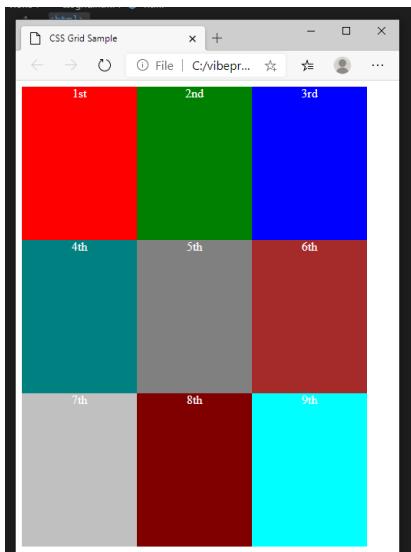
The line

```
grid-template-rows: 200px 200px 200px;
```

means we are declaring three rows, each row with a height of 200 pixels. This way, we have defined a three by three grid.

Do not compile and run. Simply double-click the file to see how it looks on the browser.

First, make the browser small.



Then expand the browser.

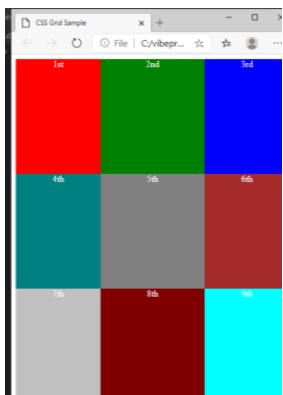


You see that even when you resize the browser, the blocks remain the same sizes.

Now let's change the CSS to make the center column expandable.

```
.container
{
    margin: 0 auto;
    width: 100%;
    height: 100%;
    display: grid;
    grid-template-columns: 150px auto 150px;
    grid-template-rows: 200px 200px 200px;
}
```

Refresh the browser. When you expand or contract the browser, the middle column expands automatically.

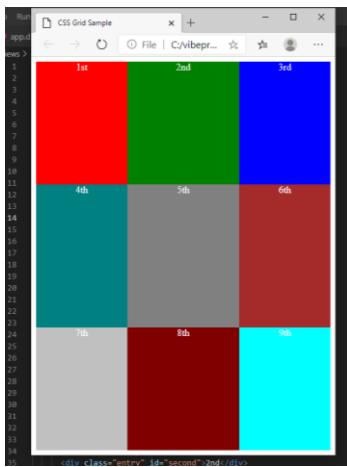


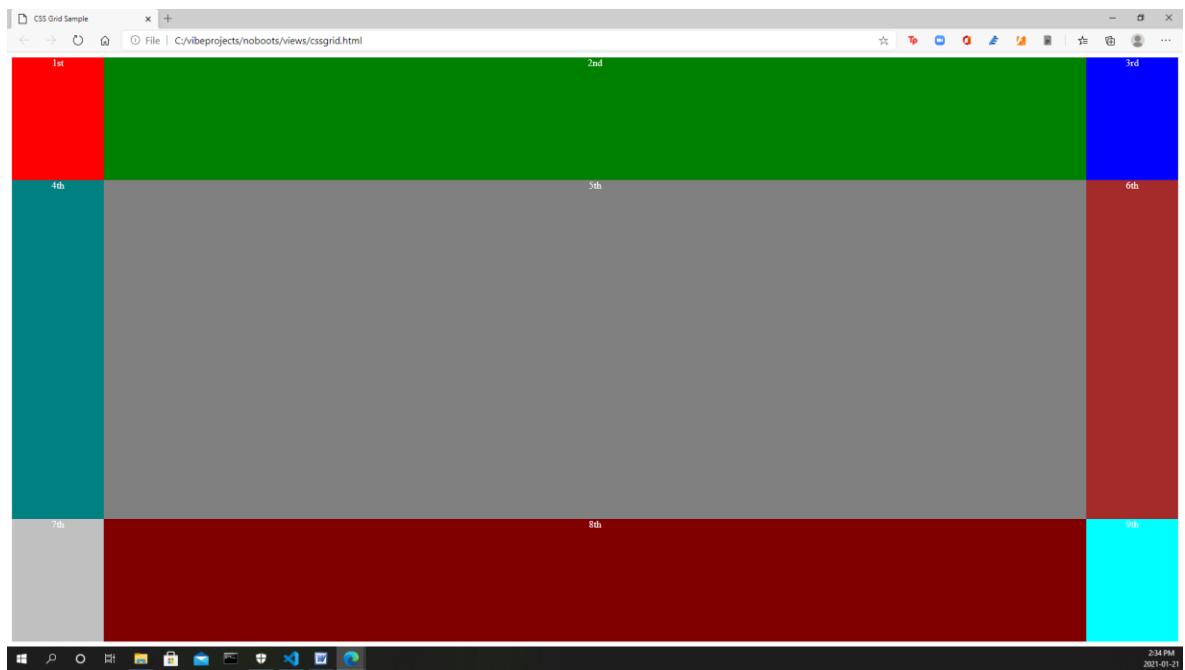


Now let's make the middle row expandable too.

```
display: grid;
grid-template-columns: 150px auto 150px;
grid-template-rows: 200px auto 200px;
```

Refresh and resize the browser.





Now both the middle column and the middle row are expandable.

This is just skimming the surface of CSS Grid. At least we are now armed with enough knowledge to develop the sticky navbar and the sticky footer without Bootstrap.

A sticky navbar and footer without Bootstrap

It is nice to work with Bootstrap. CSS becomes less intimidating and approachable. They also keep improving Bootstrap every year. However, there are also developments in standard CSS which we tend to ignore because of the familiarity and convenience of Bootstrap.

Let's create a new project with a sticky navbar and footer without using Bootstrap.

Create a project named noboots.

```
c:\vibeprojects>dub init noboots -t vibe.d
Package recipe format (sdl/json) [json]:
Name [noboots]:
Description [A simple vibe.d server application.]:
Author name [rey]:
License [proprietary]:
Copyright string [Copyright © 2021, rey]:
Add dependency (leave empty to skip) []:
Successfully created an empty project in 'c:\vibeprojects\noboots'.
Package successfully created in noboots
```

```
c:\vibeprojects>cd noboots
```

```
c:\vibeprojects\noboots>
```

Open the new directory in VS Code and edit source\app.d to make it look like this:

```
import vibe.vibe;

void main()
{
    auto router = new URLRouter;
    router.get("*", serveStaticFiles("public/"));
    router.get("/", staticTemplate!"home.dt");
    router.get("/about", staticTemplate!"about.dt");
    router.get("/contact", staticTemplate!"contact.dt");
    router.get("/login", staticTemplate!"login.dt");

    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = ["::1", "127.0.0.1"];
    listenHTTP(settings, router);
    runApplication();
}
```

Next, create views\layout.dt.

```
html
head
    title Employee Records System
    include cssjs.dt
body
    div.container
        div.menu
            include menu.dt
        div.content
            block changingcontent
        div.footer
            div.copyright Copyright &copy; The Lorem Ipsum Company 2021
```

Then create views\cssjs.dt.

```
:css
body
{
    margin: 0;
    padding: 0;
}
.container
{
    display: grid;
    grid-template-rows: 40px auto 30px;
    height: 100%;
    width: 100%;
    margin: 0;
    padding: 0;
}
/*menu styles***** */
.menu
{
    position: fixed;
    top: 0;
    width: 100%;
    padding: 10px 0;
    background-color: whitesmoke;
}
.menu-item
{
```

```
    display: inline;
}
.item-link
{
    margin-left: 30px;
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    font-size: 18px;
    color: teal;
    text-decoration: none;
}
.item-link-right
{
    float: right;
    margin-right: 30px;
}
/*end of menu styles*/
/*content styles******/
.content
{
    padding: 40px 30px;
}
/*end of content styles*/
/*footer styles******/
.footer
{
    position: fixed;
    bottom: 0;
    width: 100%;
    background-color: whitesmoke;
    padding: 5px 0;
}
.copyright
{
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    font-size: 14px;
    text-align: center;
    color: teal;
}
.form-hidden
{
    padding: 0;
    margin: 0;
    display: inline;
}
```

Then create views\menu.dt.

```
div.menu-item
  a.item-link(href="/") Home
div.menu-item
  a.item-link(href="about") About us
div.menu-item
  a.item-link(href="contact") Contact us
div.menu-item
  a.item-link.item-link-right(href="login") Login
```

Then create views\home.dt.

```
extends layout
block changingcontent
  h2 The Lorem Ipsum Company
  div.
    For the source text, <a href="https://lipsum.com/">go here.</a>
    <br /><br />
    What is Lorem Ipsum?<br /><br />
    Lorem Ipsum is simply dummy text of the printing and typesetting
    industry. Lorem Ipsum has been the industry's standard dummy
    text ever since the 1500s, when an unknown printer took a galley
    of type and scrambled it to make a type specimen book. It has
    survived not only five centuries, but also the leap into
    electronic typesetting, remaining essentially unchanged. It was
    popularised in the 1960s with the release of Letraset sheets
    containing Lorem Ipsum passages, and more recently with desktop
    publishing software like Aldus PageMaker including versions of
    Lorem Ipsum.
    <br /><br />
```

And make stub pages for views\about.dt,

```
extends layout
block changingcontent
  h2 The About Us page
  div.
    <h3>This is the about us page.</h3>
```

views\contact.dt,

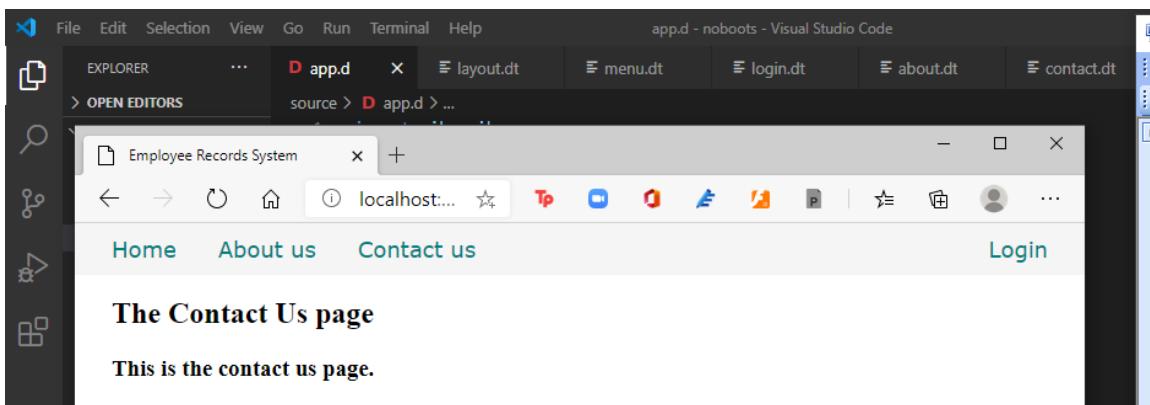
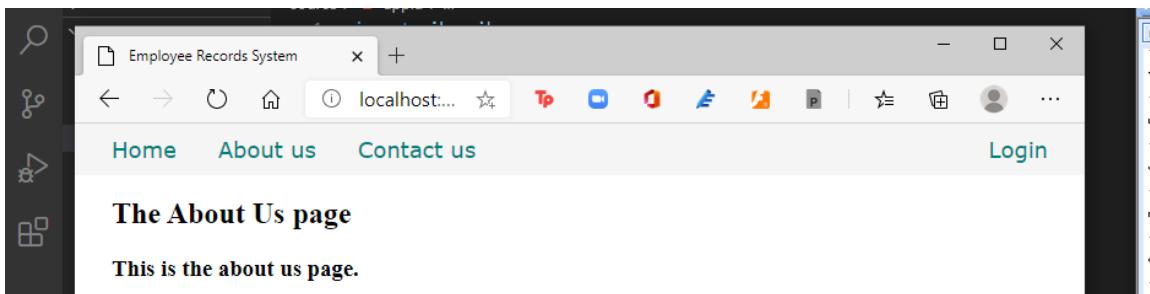
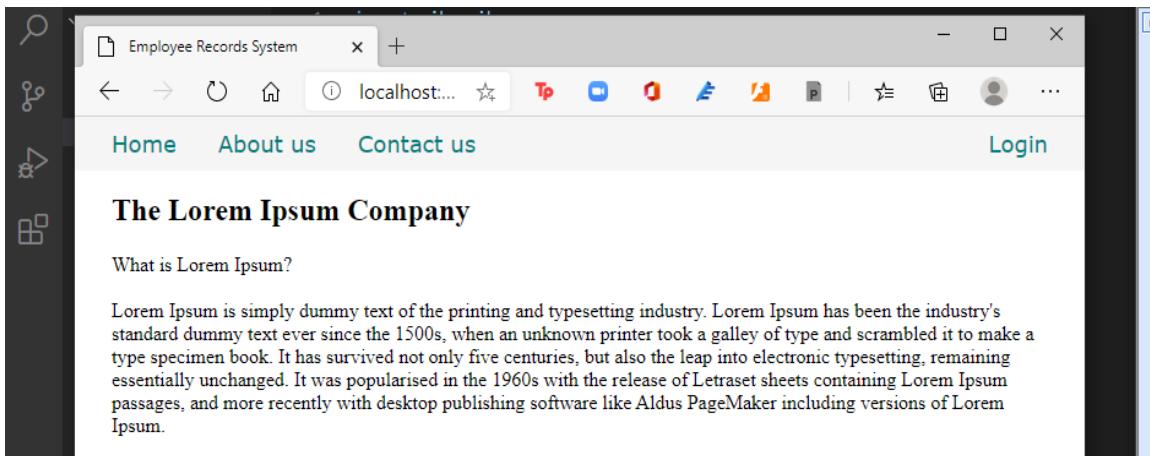
```
extends layout
block changingcontent
  h2 The Contact Us page
```

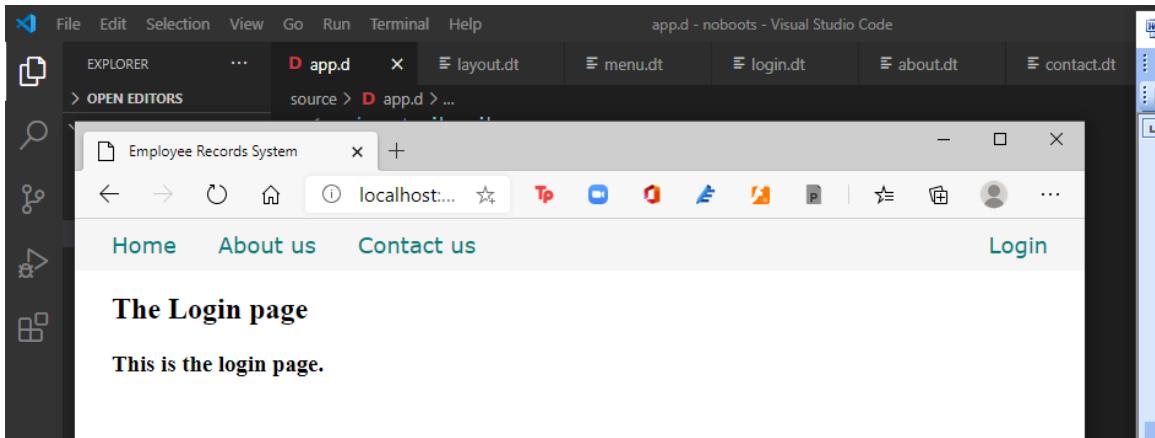
```
div.  
<h3>This is the contact us page.</h3>
```

and views\login.dt.

```
extends layout  
block changingcontent  
    h2 The Login page  
    div.  
        <h3>This is the login page.</h3>
```

After you compile and run, you should be able to see the following pages:





In the views\cssjs.dt file, we declared this:

```
.container
{
    display: grid;
    grid-template-rows: 40px auto 30px;
    height: 100%;
    width: 100%;
    margin: 0;
    padding: 0;
}
```

Which means we only declared three rows and no columns using CSS Grid, effectively making the whole page one column with three rows. The menu bar is on the top row and the footer is on the bottom row, with the middle row expanding and contracting to fit the page height.

So we did a sticky navbar and a sticky footer without Bootstrap.

Next, a modal dialogue without javascript.

Adding a modal dialogue to the menu

With CSS3, we can make modal dialogues that are actually part of the page but is hidden by default then becomes visible and takes the focus after an event.

Let us implement the Login link with a modal form.

Edit the views\menu.dt file.

```
div.menu-item
  a.item-link(href="/") Home
div.menu-item
  a.item-link(href="about") About us
div.menu-item
  a.item-link(href="contact") Contact us
div.menu-item
  a.item-link.item-link-right(href="#login") Login
div#login.modal-form
  div.modal-form-wrapper-login
    form.modal-form-grid(method="post", action="login")
      input.text-
control(name="email", type="email", placeholder=" email address")
      input.text-
control(name="password", type="password", placeholder=" password")
      input#but-reset(type="reset", value="Clear")
      input#but-submit(type="submit", value="Submit")
    a.close(href="#close") Cancel
```

We made a div with an id of #login so the link can point to it.

Add this to the end of views\cssjs.dt.

```
.modal-form
{
  position: fixed;
  font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  top: 0;
  right: 0;
  bottom: 0;
  left: 0;
  background: rgba(0,0,0,0.5);
  z-index: 99999;
  opacity: 0;
  pointer-events: none;
```

```
}

#login:target, #one_employee:target
{
    opacity: 1;
    pointer-events: auto;
}

.modal-form-grid {
    display: grid;
    grid-template: 30px 30px 30px / 1fr 1fr;
    grid-gap: 10px;
}

.text-control
{
    grid-column: 1 / 3;
}

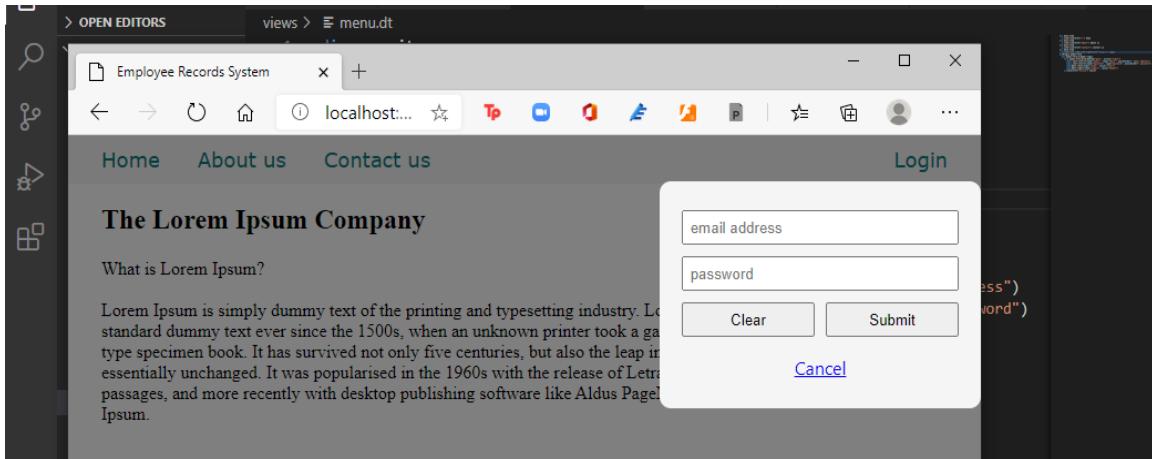
#but-reset
{
    grid-column: 1 / 2;
}

#but-submit
{
    grid-column: 2 / 3;
}

.modal-form-wrapper-login
{
    width: 250px;
    position: absolute;
    right: 0;
    margin-top: 40px;
    padding: 25px 20px;
    border-radius: 10px;
    text-align: center;
    background-color: whitesmoke;
}

.modal-form-wrapper-one_employee
{
    width: 250px;
    position: relative;
    left: 280px;
    margin-top: 40px;
    padding: 25px 20px;
    border-radius: 10px;
    text-align: center;
    background-color: whitesmoke;
}
```

After you compile and run, click on the Login link and see the modal form in action:



Using CSS Grid, you notice that all the layout terms were declared in the CSS file so the HTML is cleaner.

Let us edit the menu to make it more relevant to our final project.

Edit menu.dt again to add the modal form for the #find_employee link

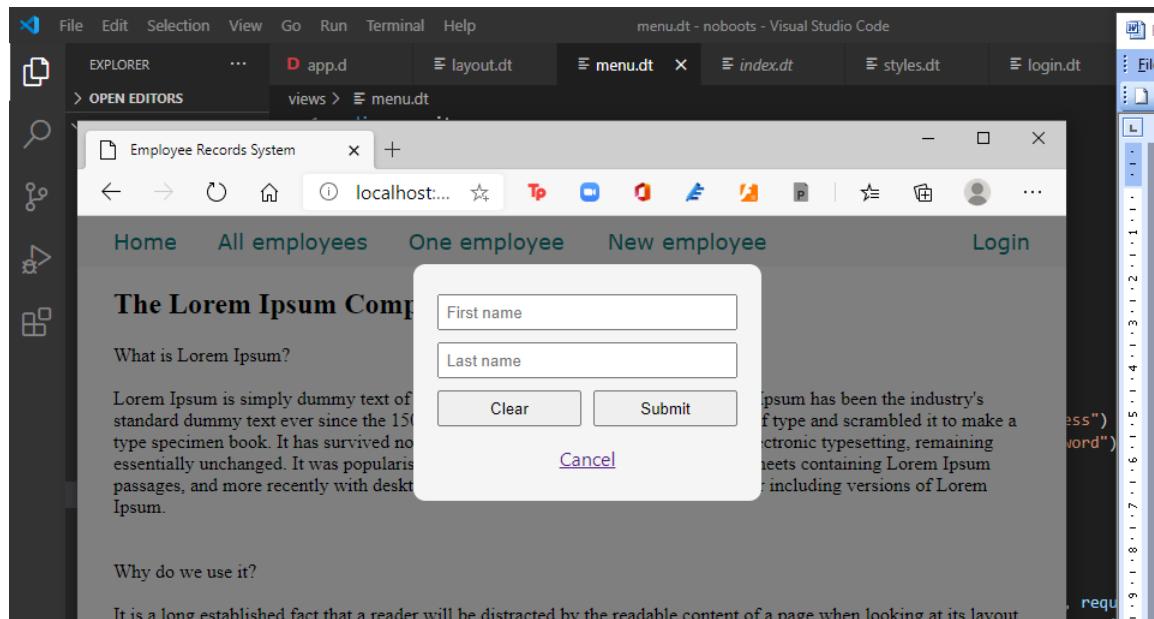
```
div.menu-item
  a.item-link(href="/") Home
div.menu-item
  a.item-link(href="all_employees") All employees
div.menu-item
  a.item-link(href="#find_employee") One employee
div.menu-item
  a.item-link(href="new_employee") New employee
div.menu-item
  a.item-link.item-link-right(href="#login") Login
div#login.modal-form
  div.modal-form-wrapper-login
    form.modal-form-grid(method="post", action="login")
      input.text-
control(name="email", type="email", placeholder=" email address")
      input.text-
control(name="password", type="password", placeholder=" password")
      input#but-reset(type="reset", value="Clear")
      input#but-submit(type="submit", value="Submit")
      a.close(href="#close") Cancel
div#one_employee.modal-form
  div.modal-form-wrapper-one_employee
```

```

form.modal-form-grid(method="post", action="find_employee")
    input.text-
control(name="fname", type="text", placeholder=" First name", required)
    input.text-
control(name="lname", type="text", placeholder=" Last name", required)
    input#but-reset(type="reset", value="Clear")
    input#but-submit(type="submit", value="Submit")
a.close(href="#close") Cancel

```

Compile and run, then refresh the browser. Click on the One employee link.



Now we also have a modal dialogue to find an employee.

That's it for elementary design. I'm no expert on web design or CSS and I'm sure your company has someone good with those.

The web framework

The web framework makes routing simpler and more intuitive as it combines business logic and displaying web pages easily. Instead of declaring all the routes in the router itself, the web framework makes use of conventions based on REST (Representational State Transfer) to match URLs to actions or routes combined with the business logic.

The five actions / requests in REST are

GET - retrieve something from the server
POST - create something new in the server
PUT - update fully something in the server
DELETE – delete something from the server
PATCH – update partially something in the server

For our purposes, we will focus on the GET and POST operations. We will use GET for simple retrieval operations and use POST for the rest. For example, if the request is only asking for a page to be displayed, we usually use the GET method. If the request is asking for data to be added, retrieved, modified or deleted, we will use the POST method.

When a form is submitted, a GET method shows the input variables of the form at the end of the URL, called the query string. With a POST method, they are not displayed. Also, a query string has a limit in length while a POST method does not have this limitation.

We don't want to change or erase what we have done so far so we can preserve this point in time and can go back to this state at any time, so we are going to create a new project and simply copy all the \views* files in this project to that new project.

Create a new project named empapp.

Press Ctrl-C to stop the running program (twice if you have to).

```
[main----) INF] Listening for requests on http://[::1]:8080/  
[main----) INF] Listening for requests on http://127.0.0.1:8080/  
[00000000----) INF] Received signal 2. Shutting down.  
Warnin^g: C6 socket ha  
ndc:\vibeprojects\noboots>les leaked at driver shutdown.  
Warning: 6 socket handles leaked at driver shutdown.
```

c:\vibeprojects\noboots>cd ..

```
c:\vibeprojects>dub init empapp -t vibe.d  
Package recipe format (sdl/json) [json]:  
Name [empapp]:
```

Description [A simple vibe.d server application.]:

Author name [rey]:

License [proprietary]:

Copyright string [Copyright © 2021, rey]:

Add dependency (leave empty to skip) []:

Successfully created an empty project in 'c:\vibeprojects\empapp'.

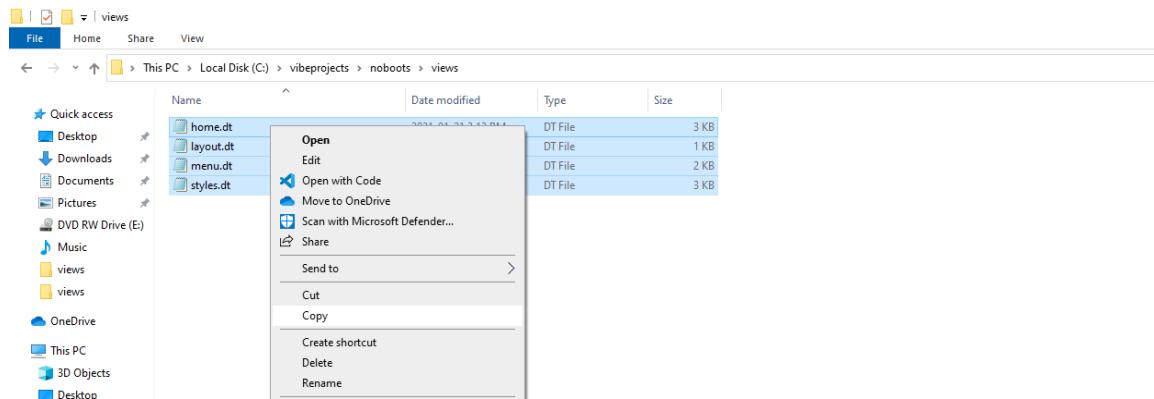
Package successfully created in empapp

c:\vibeprojects>

c:\vibeprojects>cd empapp

c:\vibeprojects\empapp>

Simply copy the files in noboots\views\ into the current project empapp\views so we don't have to create them again.



Open the new empapp folder in VS Code and edit source\app.d to make use of the web framework.

```
import vibe.vibe;
import empinterface;

void main()
{
    auto router = new URLRouter;
    router.get("*", serveStaticFiles("public/"));
    router.registerWebInterface(new EmployeeInterface);

    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = ["::1", "127.0.0.1"];

    listenHTTP(settings, router);
}
```

```
    runApplication();
}
```

The line

```
    router.registerWebInterface(new EmployeeInterface);
```

brings in the web framework. The registerWebInterface() method of the router is the facility that brings in all the goodies of the web framework. Here, the router is declaring that the class EmployeeInterface, which we have not created yet, will have all the facilities of the web framework.

We haven't created the EmployeeInterface class yet, so let's rectify that. We indicated that we are importing the empinterface module, so let's create that file.

Create source\empinterface.d with this content:

```
module empinterface;

import vibe.vibe;

class EmployeeInterface
{
    void index()
    {
        render! "home.dt";
    }

    void getFindEmployee()
    {
        response.writeBody("This is getFindEmployee");
    }
}
```

So instead of creating separate methods, we can group them all together inside one class. We don't have to state the `HTTPServerRequest` and the `HTTPServerResponse` objects as arguments for each and every method we define here because they are already available to us if we need them as the variables `request` and `response`.

The `index()` method corresponds to the root URL

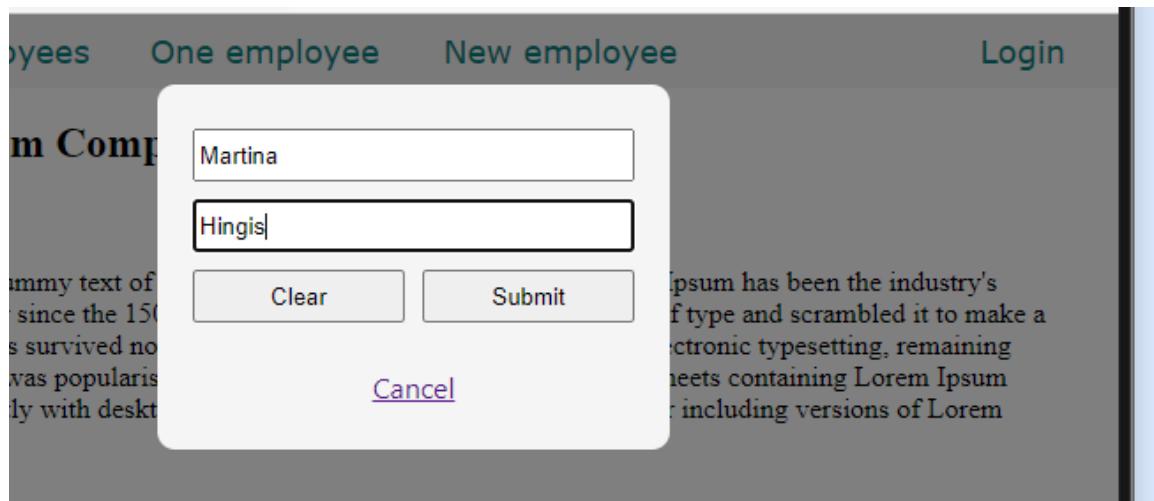
`http://localhost:8080/`

The `getFindEmployee()` method corresponds to the URL

http://localhost:8080/find_employee

which is in the modal form in views\menu.dt, with a HTTPMethod of GET:

```
div#one_employee.modal-form
  div.modal-form-wrapper-one_employee
    form.modal-form-grid(method="get", action="find_employee")
      input.text-
control(name="fname", type="text", placeholder=" First name", required)
      input.text-
control(name="lname", type="text", placeholder=" Last name", required)
      input#but-reset(type="reset", value="Clear")
      input#but-submit(type="submit", value="Submit")
```



So if we make a postLogin() method, it will correspond to the URL

http://localhost:8080/login

of the login modal form, the HTTPMethod of which is POST:

```
div#login.modal-form
  div.modal-form-wrapper-login
    form.modal-form-grid(method="post", action="login")
      input.text-
control(name="email", type="email", placeholder=" email address", required)
      input.text-
control(name="password", type="password", placeholder=" password", required)
      input#but-reset(type="reset", value="Clear")
      input#but-submit(type="submit", value="Submit")
```

The screenshot shows a web application interface. At the top, there is a navigation bar with links for 'One employee' and 'New employee'. To the right of the navigation bar is a 'Login' button. Below the navigation bar, the word 'Company' is displayed in large, bold, dark text. To the right of 'Company', there is a large, semi-transparent rectangular overlay containing a login form. The form includes two input fields: one for email ('anna@kuornikova.mail.com') and one for password ('.....'). Below the password field is a visibility icon (an eye symbol). Underneath the input fields are three buttons: 'Clear', 'Submit', and 'Cancel'. The background of the page is a light gray color with some faint text visible, which appears to be a history of the printing industry. On the far right edge of the screen, there is a vertical scroll bar with numerical tick marks from 1 to 10.

anna@kuornikova.mail.com

.....

Clear Submit Cancel

The methods we defined in the `EmployeeInterface` class (you can give the class any name), are merely stubs for now just to demonstrate the use of the web framework.

Using a web interface makes it easier to combine business logic with routing to make a full application. You can mix business logic with displaying a page, such as accessing a database before displaying the list of employees. That is what we will do next, but first we have to set up our data source.

The data source

Presently there are two popular options for a database backend. We can use MySQL, which is the most popular open-source DBMS, or we could use MongoDB, the most popular NoSQL open-source document store.

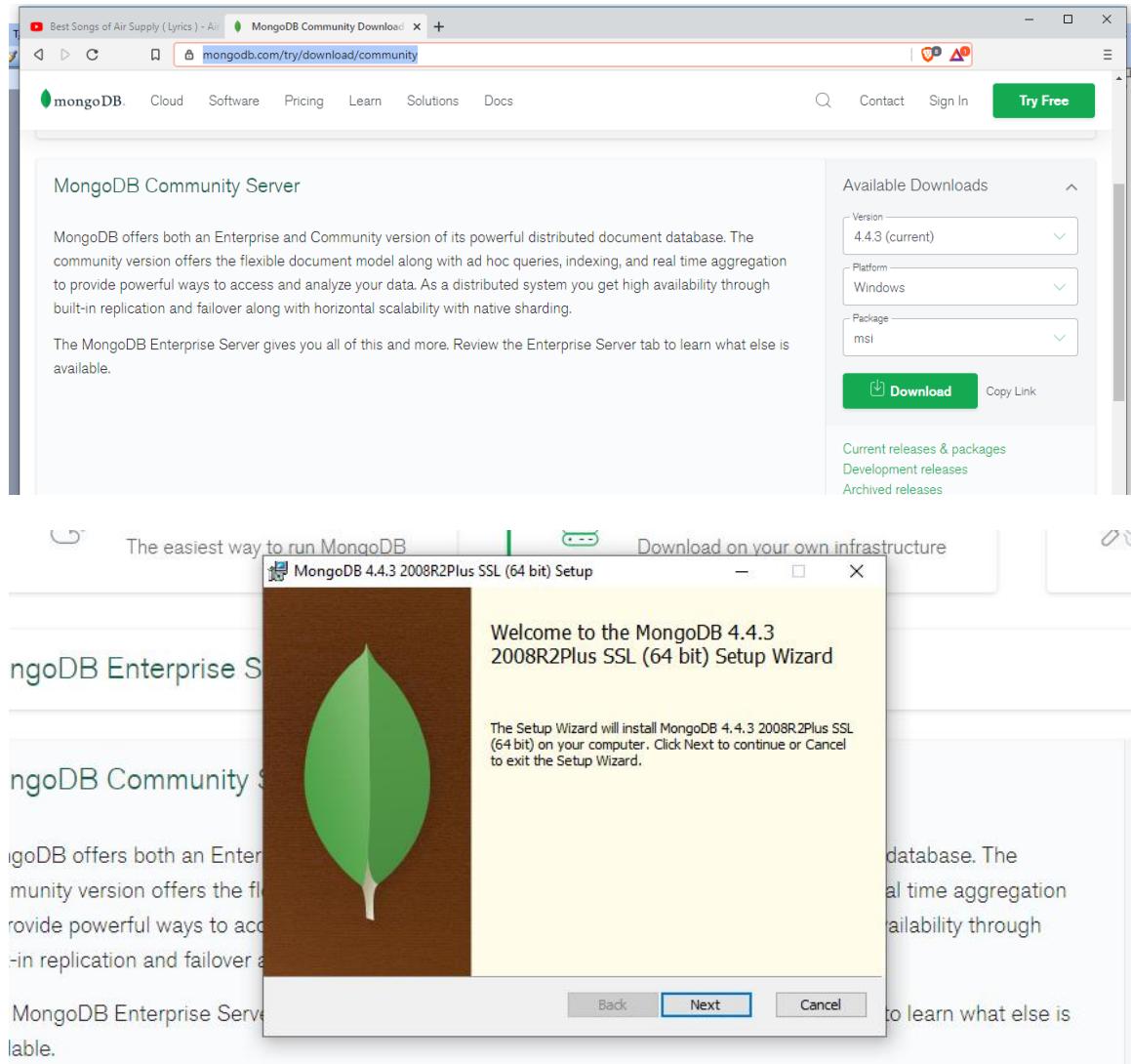
With MySQL, we have to use a separate driver that is being maintained outside of Vibe.d. With MongoDB, however, we do not need any external library as it is built-in inside Vibe.d. So for now we will use MongoDB, then simply port the completed app to MySQL.

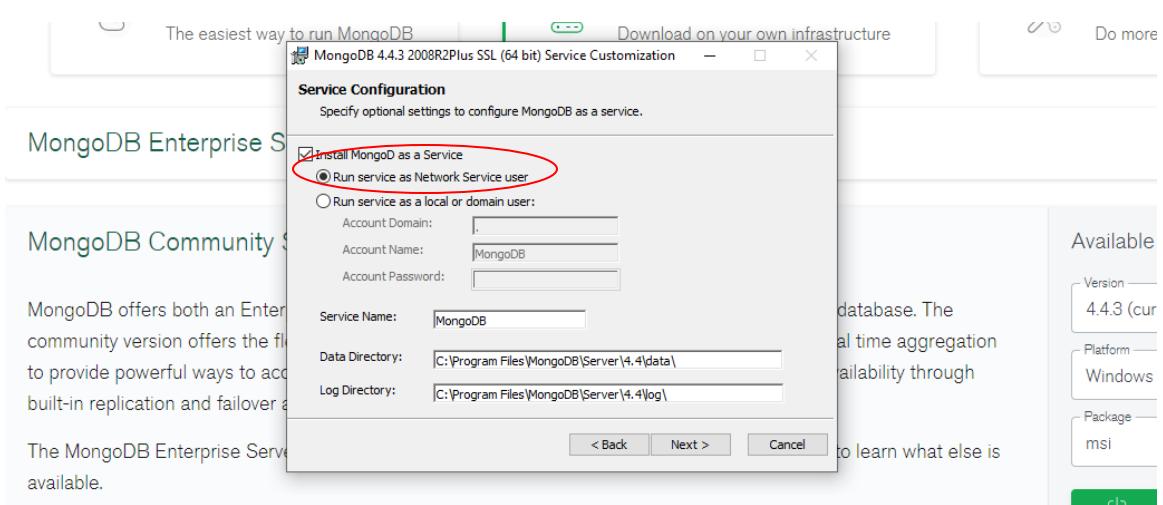
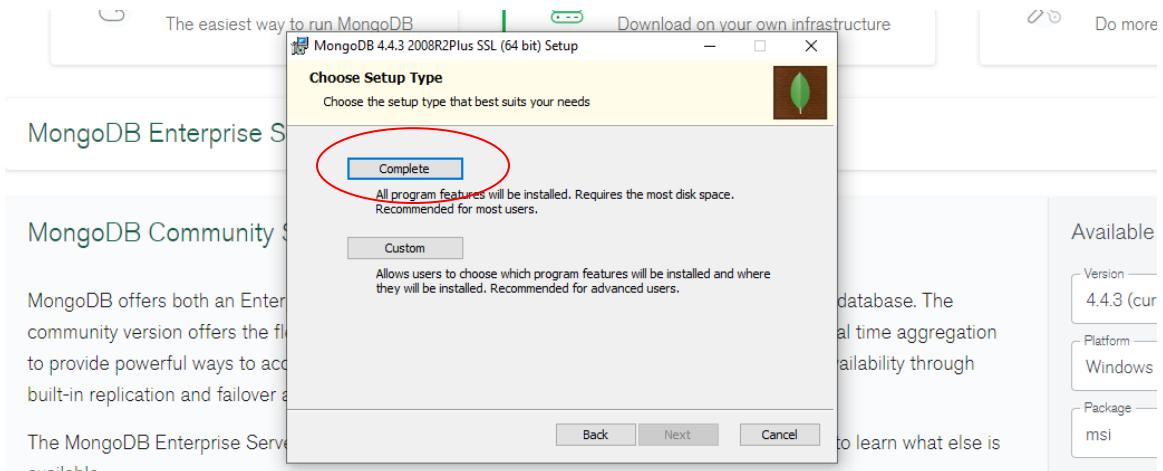
Installing MongoDB

To download the MongoDB community server, head to

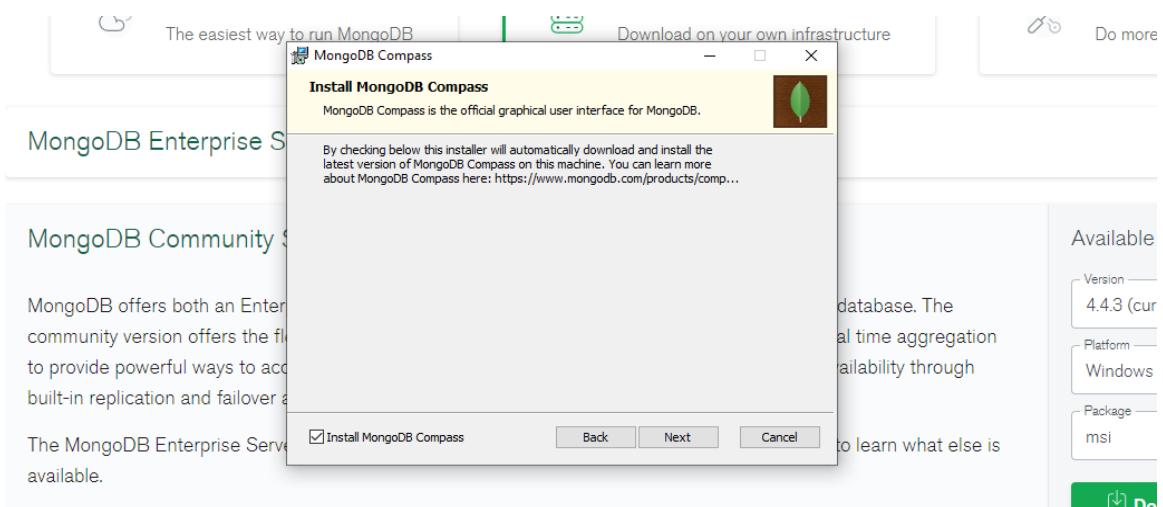
<https://www.mongodb.com/try/download/community>

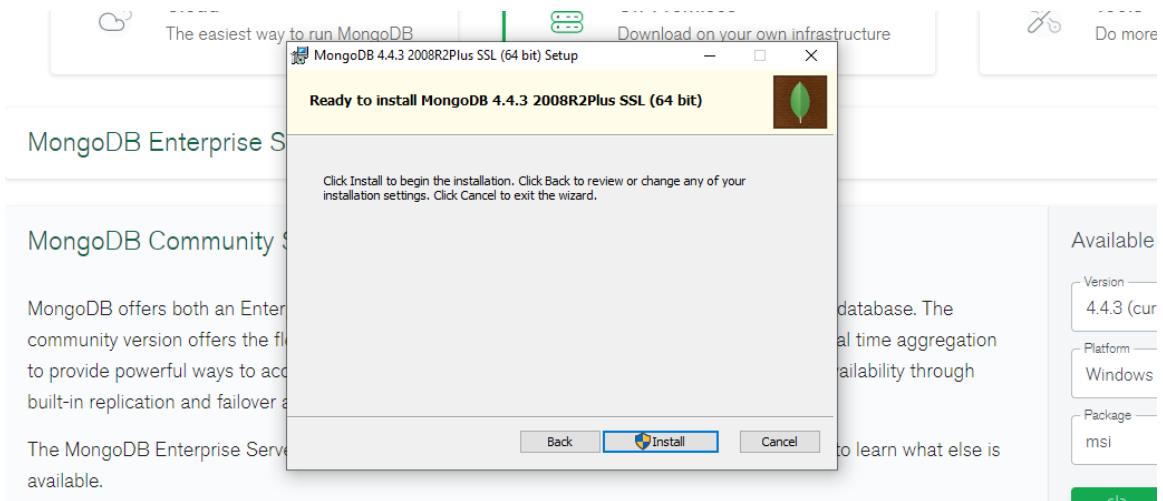
then run the downloaded file and follow the instructions.



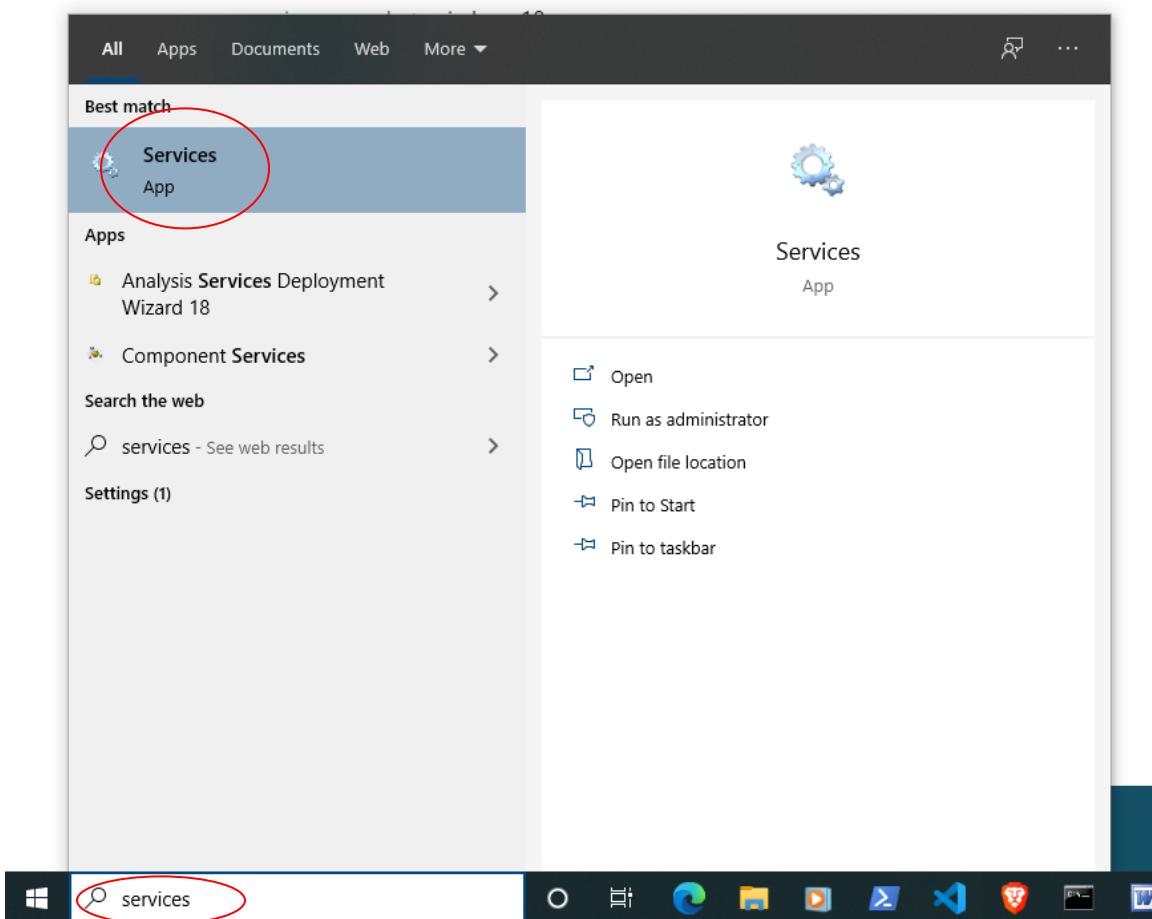


You should also install MongoDB Compass as it is convenient.

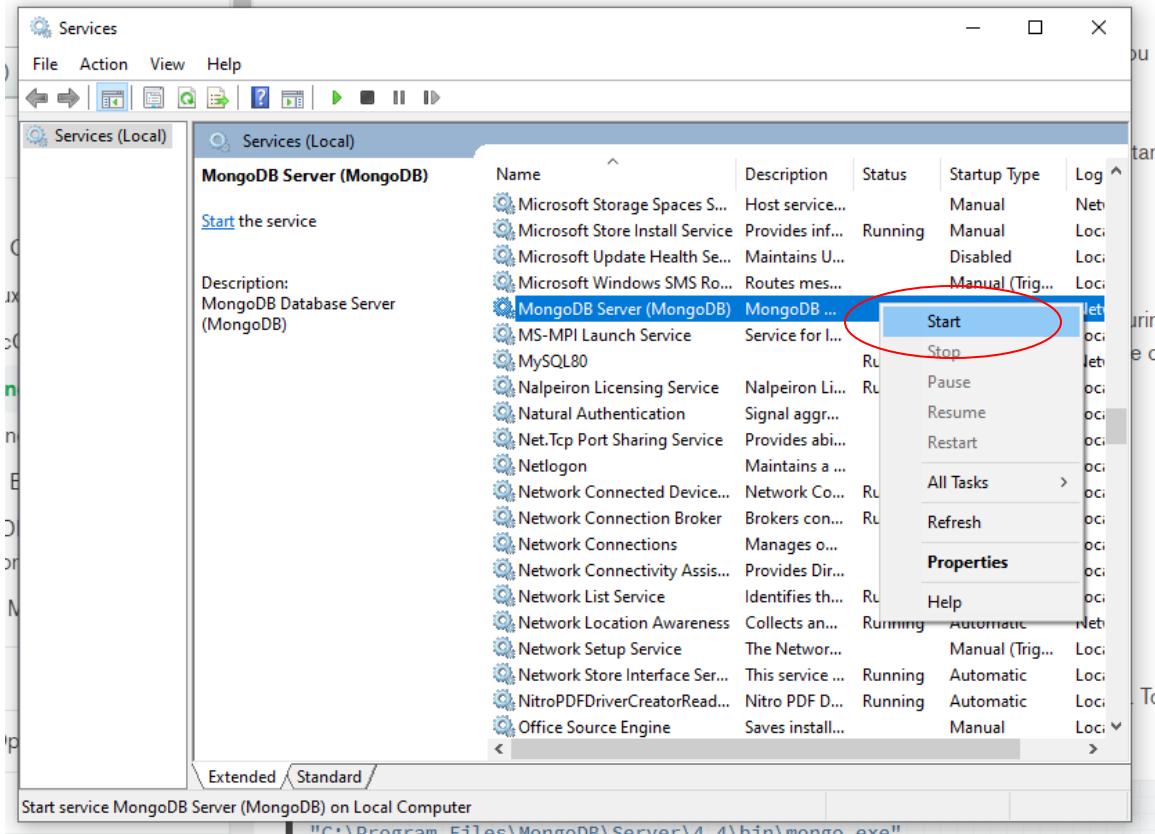




After installation of the MongoDB server, type ‘services’ on the Windows Search bar to open the Services window:



Inside the Services window, look for the MongoDB Server (MongoDB), right-click on it and click ‘Start’ to start the MongoDB server.



To test the MongoDB installation, on the terminal window, navigate to C:\Program Files\MongoDB\Server\4.4\bin.

```
C:\vibeprojects\hello>cd C:\Program Files\MongoDB\Server\4.4\bin
```

Then run the MongoDB console, mongo.exe:

```
C:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
MongoDB shell version v4.4.3
connecting
to:
mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("9b423cb4-8414-4f73-a169-71db6fa33c76") }
MongoDB server version: 4.4.3
---
```

The server generated these startup warnings when booting:

```
2021-02-02T14:57:50.217-05:00: Access control is not enabled for the database.
Read and write access to data and configuration is unrestricted
---
```

Enable MongoDB's free cloud-based monitoring service, which will then receive and display metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to you and anyone you share the URL with. MongoDB may use this information to make product improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()

To permanently disable this reminder, run the following command:
db.disableFreeMonitoring()

>

Inside the MongoDB terminal, type show databases.

```
> show databases
admin 0.000GB
config 0.000GB
local 0.000GB
>
```

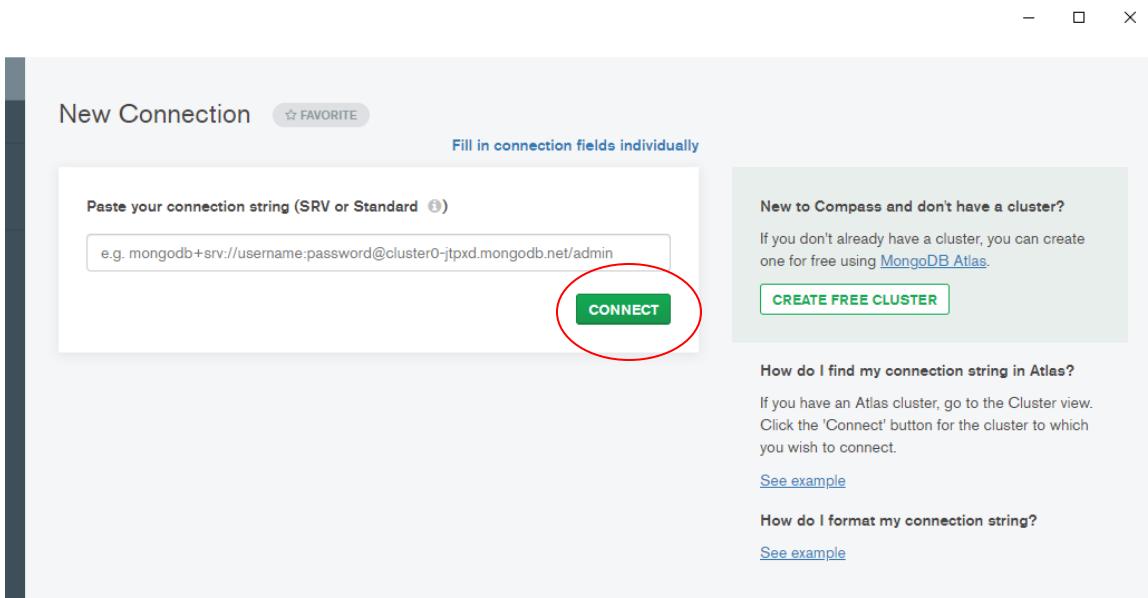
To exit from the MongoDB console, type quit().

```
> quit()
```

C:\Program Files\MongoDB\Server\4.4\bin>

MongoDB Compass

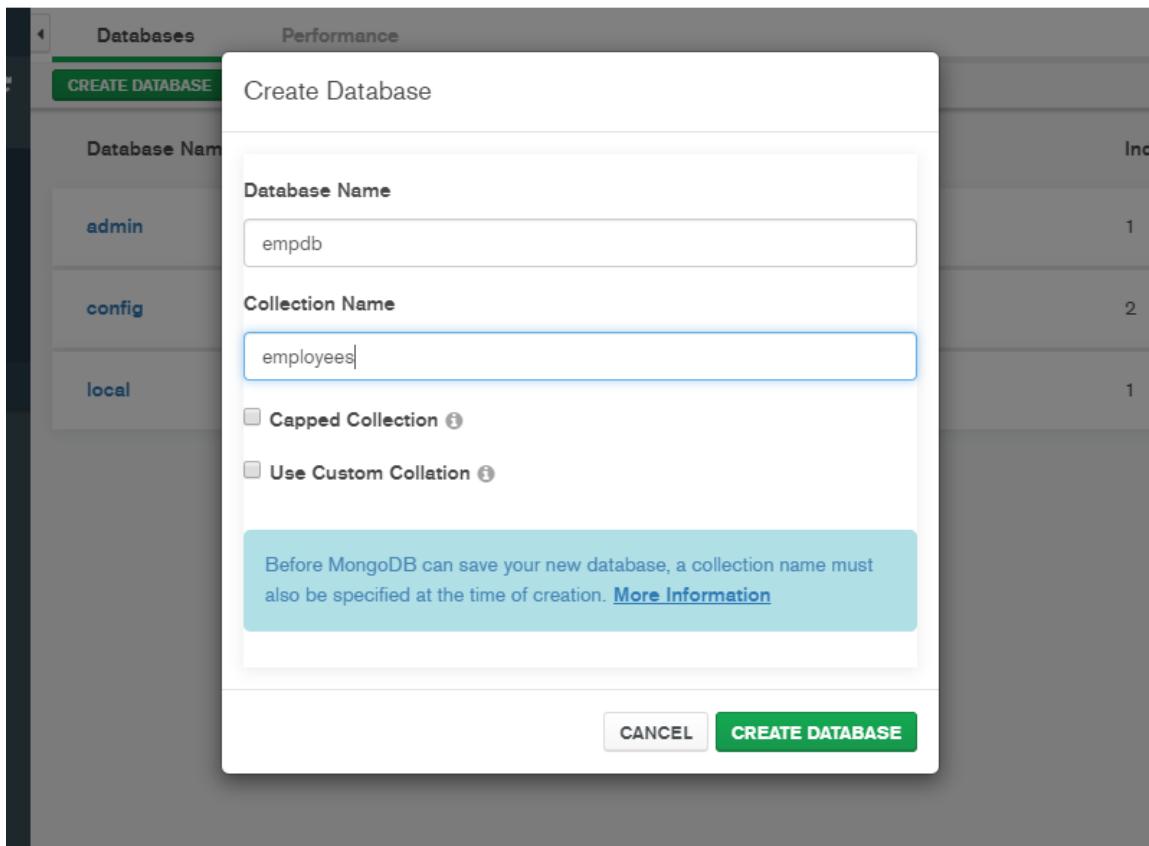
Aside from the MongoDB console, we can also use the MongoDB Compass app to create and manage databases, collections and records.



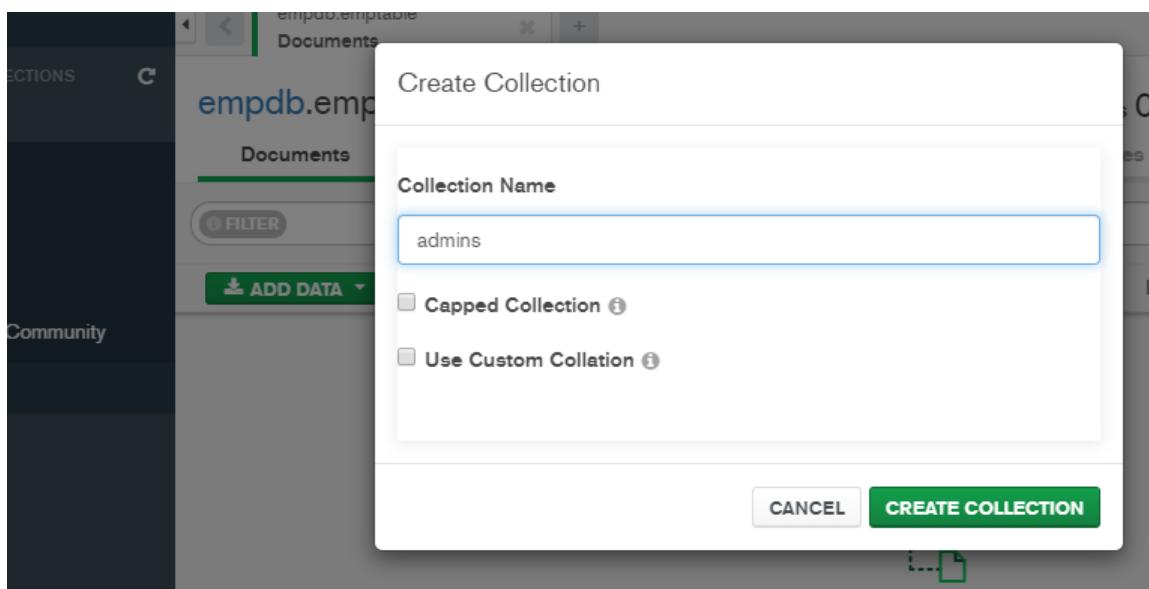
Create a database named empdb with a collection named employees.

The screenshot shows the main interface of MongoDB Compass. On the left is a sidebar with 'Local' selected, showing '3 DBS' and '1 COLLECTIONS'. Below this are sections for 'HOST' (localhost:27017), 'CLUSTER' (Standalone), and 'EDITION' (MongoDB 4.4.3 Community). A 'Filter your data' search bar is at the bottom of the sidebar. The main area has tabs for 'Databases' (which is active and has a red circle around its tab) and 'Performance'. The 'Databases' tab displays a table with three rows:

Database Name	Storage Size	Collections
admin	32.0KB	0
config	32.0KB	0
local	36.0KB	1



After creating the database and collection, create another collection in empdb called admins.



Then, add one or two records to the admins collection.

The screenshot shows the MongoDB Compass interface. On the left, the sidebar lists databases (Local, HOST localhost:27017, CLUSTER Standalone) and collections (admin, config, empdb). A red circle highlights the 'admins' collection under 'empdb'. In the main area, the 'empdb.admins' collection is selected. A red circle highlights the 'ADD DATA' button, which has a dropdown menu open with 'Insert Document' selected. The status bar at the bottom right says 'Displaying documents'.

The screenshot shows the 'Insert to Collection empdb.admins' dialog box. It contains a JSON document:

```

1 /**
2  * Paste one or more documents here
3 */
4 {
5   "_id": {
6     "$oid": "601990d6c839a725a275c115"
7   },
8   "email": "admin2@lorem.com",
9   "password": "secret"
10 }

```

At the bottom right of the dialog box, there are 'CANCEL' and 'INSERT' buttons. The 'INSERT' button is highlighted with a red circle.

We can also create an index on the email field of the emptable collection using Compass but for variety's sake let's try the MongoDB console to do that, mongo.exe.

Using mongo.exe, which is in C:\Program Files\MongoDB\Server\4.4\bin\, create a unique index on the email field of the emptable collection.

```

c:\Program Files\MongoDB\Server\4.4\bin>mongo.exe
MongoDB shell version v4.4.3
connecting
to:
mongodb://127.0.0.1:27017/?compressors=disabled&gssapiServiceName=mongodb
Implicit session: session { "id" : UUID("98e129ff-5e12-4828-b44a-6dceba2875") }
MongoDB server version: 4.4.3
Welcome to the MongoDB shell.

---
---
---
---
---

> use empdb
switched to db empdb
> db.admins.createIndex({email: 1}, {unique: true})
{
    "createdCollectionAutomatically" : false,
    "numIndexesBefore" : 1,
    "numIndexesAfter" : 2,
    "ok" : 1
}
> quit()

```

c:\Program Files\MongoDB\Server\4.4\bin>

When you look at MongoDB Compass again, you will see that the index was created.

The screenshot shows the MongoDB Compass interface connected to the 'empdb' database. On the left sidebar, under 'Local', it shows 4 DBs and 2 Collections. The 'empdb.admins' collection is selected. In the main area, the 'Indexes' tab is highlighted with a red oval. Below it, the table shows one index entry:

Name	Type	Size	Usage	Properties	Drop
_id	REGULAR	4.1 KB	0 since Sat Feb 13 2021	UNIQUE	

Now our MongoDB database backend is ready.

The ‘schema’

MongoDB uses the term ‘collection’ because the ‘table’ we are used to in DBMS’s are a series of rows and columns with the fixed schema defined when you create the tables. In MongoDB, the tables are a collection of ‘documents’ with no fixed schema, where each and every document, or record, can be potentially different, which is a fundamental difference from the traditional DBMS.

The Lorem app is a simple employee records maintenance system of the imaginary Lorem Ipsum Company, where user admins can view, add, edit and delete employee records. These should cover the basic operations done on database tables and could easily be adapted and extended for use on products or services instead of personnel.

The database is named empdb. The two collections under empdb are the admins and employees collections.

For our purposes, we will use the ‘predefined schema’ below so we do not go awry with our new-found freedom.

The employees collection – containing the employee records:

email	- email address, will also be used as username, string
pword	- password, string
dept	- department name, string
paygd	- salary grade, string
fname	- firstname, string
lname	- lastname, string
phone	- phone number, string
photo	- full path to the employee photo file, string
street	- street address minus the city, string
city	- the city part of the address, string
province	- the province part of the address, string
postcode	- postal code of the address, string

The admins collection - the administrators of the employee records:

email	- email address, which serves as username, string
password	- password, string

MongoDB automatically provides an indexed `_id` field. When a new document is added and no `_id` is specified, a new `_id` is generated using a combination of timestamp and random numbers to make a unique `_id`.

Accessing a MongoDB database

To access a MongoDB database from Vibe.d, we create a MongoClient connection, then access the MongoDB object from that connection, then the particular MongoCollection can be accessed through that MongoDB object. The vibe.db.mongo.mongo module contains these classes.

MongoClient – represents a MongoDB server connection

MongoDatabase – the database we access through the MongoClient object

MongoCollection – the collection(s) we can extract from the MongoDatabase object

We simply use the connectMongoDB() method to make a connection to the MongoDB server and receive a MongoClient instance to start the ball rolling.

Edit source\app.d:

```
import vibe.vibe;
import empinterface;

void main()
{
    auto router = new URLRouter;
    router.get("*", serveStaticFiles("public/"));
    router.registerWebInterface(new EmployeeInterface);

    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = ["::1", "127.0.0.1"];

    listenHTTP(settings, router);
    runApplication();
}
```

Edit source\empinterface.d:

```
module empinterface;

import vibe.vibe;
import empmodelmongo;

class EmployeeInterface
{
    void index()
    {
        render! "home.dt";
```

```

    }

    void getFindEmployee()
    {
        response.writeBody("This is getFindEmployee");
    }
}

```

Here, we are importing the empmodelmongo module into our EmployeeInterface class. We will try to separate the business logic code from the code that talks to the database. The EmployeeModel class will contain the code that will talk to the database, leaving the EmployeeInterface class with the business logic.

Create source\empmmodelmongo.d that will contain the EmployeeModel class.

```

module empmodelmongo;

import vibe.db.mongo.mongo;

class EmployeeModel
{
    private MongoCollection emptable, admins;

    this()
    {
        MongoClient conn = connectMongoDB("localhost");
        MongoDB empdb = conn.getDatabase("empdb");
        emptable = empdb["employees"];
        admins = empdb["admins"];
    }
}

```

In D, this() is the class constructor and ~this() is the destructor. Whatever you want to initialize at the creation of the object, you put inside the constructor.

Inside the constructor of the class EmployeeModel of source\empmmodelmongo.d, we instantiated a MongoClient object named conn, then we used conn to access the database empdb, then we used empdb to access the employees and the admins collections.

Compile and run to see if there are any errors in accessing the MongoDB server.

It is easy to have errors as there are now two running servers communicating with each other: our web server and the MongoDB server.

Dealing with (some) MongoDB-related errors

There are a so ways our code can go wrong, of course. Now that we have two running servers (MongoDB and our web server), things get complicated further. Here are some of what I encountered. Sometimes the errors may be caused by the MongoDB server not running, but that can easily be resolved by starting the MongoDB server through the Windows Services app.

Here's one error.

```
empapp ~master: building configuration "application"...
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\settings.d(130,3): Error: couldn't find field __expand_field_0
of      type      string      in      Tuple("mongodb://localhost:27017/?safe=true"[27..31],
"mongodb://localhost:27017/?safe=true"[32..36])
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(58,33):                      called from here:
parseMongoDBUrl(settings, url)
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(41,7):                      called from here: this.this("mongodb://" 
~ host ~ ":" ~ to(port) ~ "/?safe=true")
source\empmodelmongo.d(11,36):          called from here: connectMongoDB("localhost")
C:\D\dmdd2\windows\bin\dmdd.exe failed with exit code 1.

C:\vibeprojects\empapp>
```

One way of resolving this is to change “localhost” to “127.0.0.1”, including the quotation marks, making it a string. At least it is telling us which part of our code elicited an error (line 11, column 36).

Sometimes the error messages do not indicate where in our code the error is.

Here is a nasty one:

```
empapp ~master: building configuration "application"...
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\settings.d(130,3): Error: couldn't find field __expand_field_0
of      type      string      in      Tuple("mongodb://localhost:27017/?safe=true"[27..31],
"mongodb://localhost:27017/?safe=true"[32..36])
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(58,33):                      called from here:
parseMongoDBUrl(settings, url)
source\empmodelmongo.d(33,38):          called from here: connectMongoDB("mongodb://localhost:27017/?safe=true")
C:\D\dmdd2\windows\bin\dmdd.exe failed with exit code 1.

C:\vibeprojects\empapp>
```

Although it is indicated which part of the code went wrong, we have no idea why. You can try replacing localhost with 127.0.0.1, but sometimes it still would'nt work. Until in your desperation you try creating a new project altogether and copying the same code, then suddenly it works. I assure you that this code runs smoothly in my Ubuntu system.

Here is another error:

```
Running .\empapp.exe

vibe.db.mongo.connection.MongoDriverException@C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-d\mongodb\vibe\db\mongo\connection.d(190): Failed to connect to
MongoDB server at localhost:27017.
-----
0x00007FF667F56E46 in vibe.db.mongo.connection.MongoConnection.connect at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\connection.d(190)
0x00007FF667F525BC in vibe.db.mongo.client.MongoClient.this._lambda2 at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(64)
0x00007FF667F553C6 in
vibe.core.connectionpool.ConnectionPool!(vibe.db.mongo.connection.MongoConnection).Conn
ectionPool.lockConnection at C:\Users\rey\AppData\Local\dub\packages\vibe-core-
1.13.0\vibe-core\source\vibe\core\connectionpool.d(94)
0x00007FF667F52B61 in vibe.db.mongo.client.MongoClient.lockConnection at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(170)
0x00007FF667F524BE in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(71)
0x00007FF667F522EB in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(41)
0x00007FF667F521E4 in vibe.db.mongo.mongo.connectMongoDB at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\mongo.d(91)
0x00007FF667F4ACE1 in empmodelmongo.EmployeeModel.this at
C:\vibeprojects\empapp\source\empmodelmongo.d(28)
0x00007FF667F4A140 in empinterface.EmployeeInterface.this at
C:\vibeprojects\empapp\source\empinterface.d(12)
0x00007FF667F22FB6 in D main at C:\vibeprojects\empapp\source\app.d(8)
0x00007FF6683FDD53 in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).runAll()._lambda1()
0x00007FF6683FDB6F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).tryExec(scope void delegate())
0x00007FF6683FDC7B in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).runAll()
0x00007FF6683FDB6F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).tryExec(scope void delegate())
0x00007FF6683FD976 in d_run_main2
0x00007FF6683C73C3 in d_run_main
0x00007FF667F25032 in app._d_cmain!().main at
C:\D\dm2\windows\bin\..\..\src\druntime\import\core\internal\entrypoint.d(29)
0x00007FF6684DE858 in __scrt_common_main_seh at
d:\agent\_work\63\s\src\vc\crt\vcstartup\src\startup\exe_common.inl(288)
0x00007FFEE4167034 in BaseThreadInitThunk
0x00007FFEE4D3D0D1 in RtlUserThreadStart
object.Exception@C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(232): Failed to connect to [0:0:0:0:0:0:1]:27017: refused
-----
0x00007FF667F24423 in std.exception.bailOut!(object.Exception).bailOut at
C:\D\dm2\windows\bin\..\..\src\phobos\std\exception.d(516)
0x00007FF667F24329 in std.exception.enforce!().enforce!bool.enforce at
C:\D\dm2\windows\bin\..\..\src\phobos\std\exception.d(437)
0x00007FF66829370D in vibe.core.net.connectTCP._lambda5 at
C:\D\dm2\windows\bin\..\..\src\phobos\std\exception.d(434)
```

```

0x00007FF668293389 in vibe.core.net.connectTCP at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(205)
0x00007FF668280941 in vibe.core.net.connectTCP at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(188)
0x00007FF667F567CA in vibe.db.mongo.connection.MongoConnection.connect at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(171)
0x00007FF667F525BC in vibe.db.mongo.client.MongoClient.this._lambda2 at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(64)
0x00007FF667F553C6 in
vibe.core.connectionpool.ConnectionPool!(vibe.db.mongo.connection.MongoConnection).Conn-
ectionPool.lockConnection at C:\Users\rey\AppData\Local\dub\packages\vibe-core-
1.13.0\vibe-core\source\vibe\core\connectionpool.d(94)
0x00007FF667F52B61 in vibe.db.mongo.client.MongoClient.lockConnection at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(170)
0x00007FF667F524BE in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(71)
0x00007FF667F522EB in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\client.d(41)
0x00007FF667F521E4 in vibe.db.mongo.mongo.connectMongoDB at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.2\vibe-
d\mongodb\vibe\db\mongo\mongo.d(91)
0x00007FF667F4ACE1 in empmodelmongo.EmployeeModel.this at
C:\vibeprojects\empapp\source\empmodelmongo.d(28)
0x00007FF667F4A140 in empinterface.EmployeeInterface.this at
C:\vibeprojects\empapp\source\empinterface.d(12)
0x00007FF667F22FB6 in D main at C:\vibeprojects\empapp\source\app.d(8)
0x00007FF6683FDD53 in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).runAll()._lambda1())
0x00007FF6683FDB6F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).tryExec(scope void delegate()))
0x00007FF6683FDC7B in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).runAll())
0x00007FF6683FDB6F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).tryExec(scope void delegate()))
0x00007FF6683FD976 in d_run_main2
0x00007FF6683C73C3 in d_run_main
0x00007FF667F25032 in app._d_cmain!().main at
C:\D\dm2\windows\bin\..\..\src\druntime\import\core\internal\entrypoint.d(29)
0x00007FF6684DE858 in __scrt_common_main_seh at
d:\agent\_work\63\s\src\vc\crt\vcstartup\src\startup\exe_common.inl(288)
0x00007FFEE4167034 in BaseThreadInitThunk
0x00007FFEE4D3D0D1 in RtlUserThreadStart
Program exited with code 1

```

In this case, try changing localhost to 127.0.0.1. If that worked, then that might mean your hosts file does not contain an entry that maps 127.0.0.1 to localhost. You can try editing your hosts file, located in C:\Windows\System32\drivers\etc\.

```
C:\> Windows > System32 > drivers > etc > hosts
1 # Copyright (c) 1993-2009 Microsoft Corp.
2 #
3 # This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
4 #
5 # This file contains the mappings of IP addresses to host names. Each
6 # entry should be kept on an individual line. The IP address should
7 # be placed in the first column followed by the corresponding host name.
8 # The IP address and the host name should be separated by at least one
9 # space.
10 #
11 # Additionally, comments (such as these) may be inserted on individual
12 # lines or following the machine name denoted by a '#' symbol.
13 #
14 # For example:
15 #
16 #      102.54.94.97    rhino.acme.com        # source server
17 #      38.25.63.10    x.acme.com            # x client host
18 #
19 # localhost name resolution is handled within DNS itself.
20 127.0.0.1      localhost
21 ::1            localhost
22
```

Uncomment the line by deleting the hash mark (#). You will need to restart your computer for the changes to take effect.

And here is an error when your MongoDB server is not running:

```
Copying files for vibe-d:tls...
Running .\lorem.exe

vibe.db.mongo.connection.MongoDriverException@C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-d\mongodb\vibe\db\mongo\connection.d(190): Failed to connect to
MongoDB server at 127.0.0.1:27017.
-----
0x00007FF7DBA2B146      in      vibe.db.mongo.connection.MongoConnection.connect      at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\connection.d(190)
0x00007FF7DBA2999C      in      vibe.db.mongo.client.MongoClient.this.__lambda2      at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(64)
0x00007FF7DBA359E6      in      vibe.core.connectionpool.ConnectionPool!(vibe.db.mongo.connection.MongoConnection).Conn
ectionPool.lockConnection      at      C:\Users\rey\AppData\Local\dub\packages\vibe-core-
1.13.0\vibe-core\source\vibe\core\connectionpool.d(94)
0x00007FF7DBA29F41      in      vibe.db.mongo.client.MongoClient.lockConnection      at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(170)
0x00007FF7DBA2989E      in      vibe.db.mongo.client.MongoClient.this      at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(71)
```

0x00007FF7DBA296CB in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(41)
0x00007FF7DBA32F24 in vibe.db.mongo.connectMongoDB at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\mongo.d(91)
0x00007FF7DBA26A81 in empmongo.EmployeeModel.this at
C:\vibeprojects\lorem\source\empmongo.d(34)
0x00007FF7DBA24F20 in empcontrol.EmployeeController.this at
C:\vibeprojects\lorem\source\empcontrol.d(45)
0x00007FF7DB9D3D56 in D main at C:\vibeprojects\lorem\source\app.d(8)
0x00007FF7DBED8D33 in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).runAll().__lambda1()
0x00007FF7DBED8B4F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).tryExec(scope void delegate())
0x00007FF7DBED8C5B in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).runAll()
0x00007FF7DBED8B4F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][]*).tryExec(scope void delegate())
0x00007FF7DBED8956 in d_run_main2
0x00007FF7DBEA2783 in d_run_main
0x00007FF7DB9D65C2 in app._d_cmain!().main at
C:\D\mdm2\windows\bin\..\..\src\druntime\import\core\internal\entrypoint.d(29)
0x00007FF7DBFBAA558 in __scrt_common_main_seh at
d:\agent_work\63\s\src\vctools\crt\vcstartup\src\startup\exe_common.inl(288)
0x00007FFFBBF67034 in BaseThreadInitThunk
0x00007FFFC01BD241 in RtlUserThreadStart
object.Exception@C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(232): Failed to connect to 127.0.0.1:27017: refused

0x00007FF7DB9D5403 in std.exception.bailOut!(object.Exception).bailOut at
C:\D\mdm2\windows\bin\..\..\src\phobos\std\exception.d(516)
0x00007FF7DB9D5309 in std.exception.enforce!().enforce!bool.enforce at
C:\D\mdm2\windows\bin\..\..\src\phobos\std\exception.d(437)
0x00007FF7DBD66A7D in vibe.core.net.connectTCP.__lambda5 at
C:\D\mdm2\windows\bin\..\..\src\phobos\std\exception.d(434)
0x00007FF7DBD666F9 in vibe.core.net.connectTCP at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(205)
0x00007FF7DBD57211 in vibe.core.net.connectTCP at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(188)
0x00007FF7DBA2AAC in vibe.db.mongo.connection.MongoConnection.connect at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\net.d(171)
0x00007FF7DBA2999C in vibe.db.mongo.client.MongoClient.this.__lambda2 at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(64)
0x00007FF7DBA359E6 in
vibe.core.connectionpool.ConnectionPool!(vibe.db.mongo.connection.MongoConnection).Conn-
ectionPool.lockConnection at C:\Users\rey\AppData\Local\dub\packages\vibe-core-
1.13.0\vibe-core\source\vibe\core\connectionpool.d(94)
0x00007FF7DBA29F41 in vibe.db.mongo.client.MongoClient.lockConnection at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(170)
0x00007FF7DBA2989E in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(71)
0x00007FF7DBA296CB in vibe.db.mongo.client.MongoClient.this at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\client.d(41)

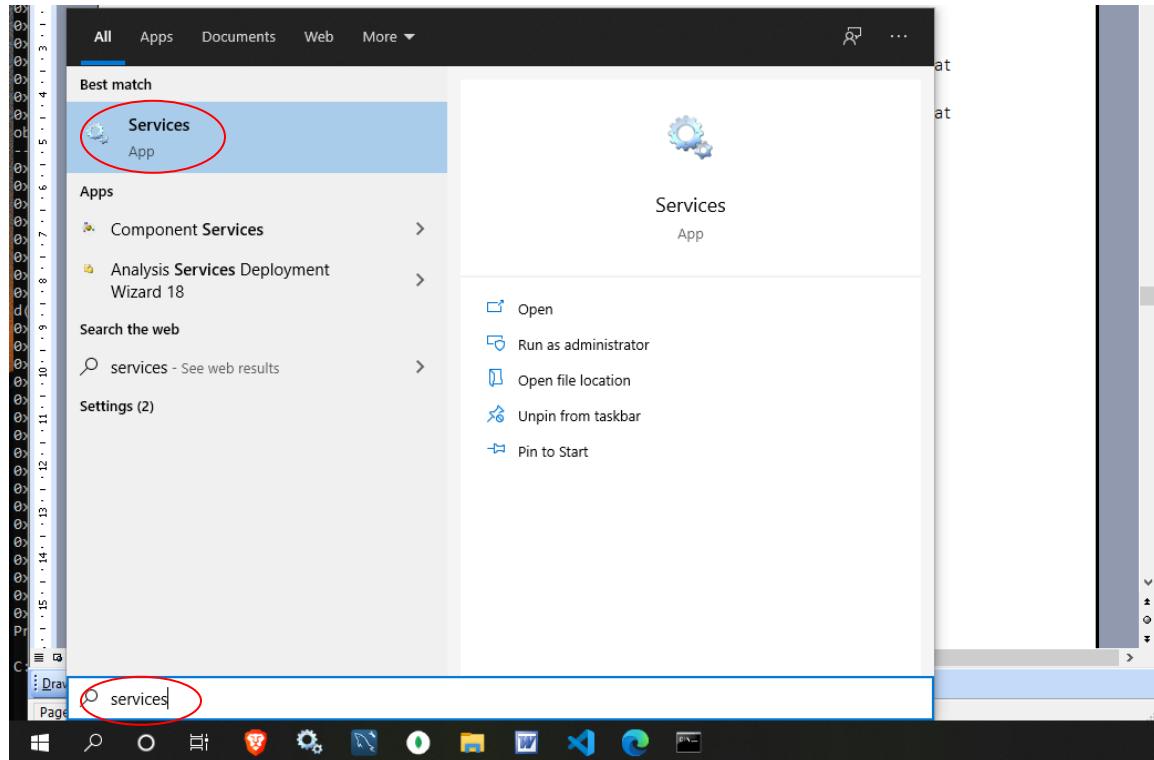
```

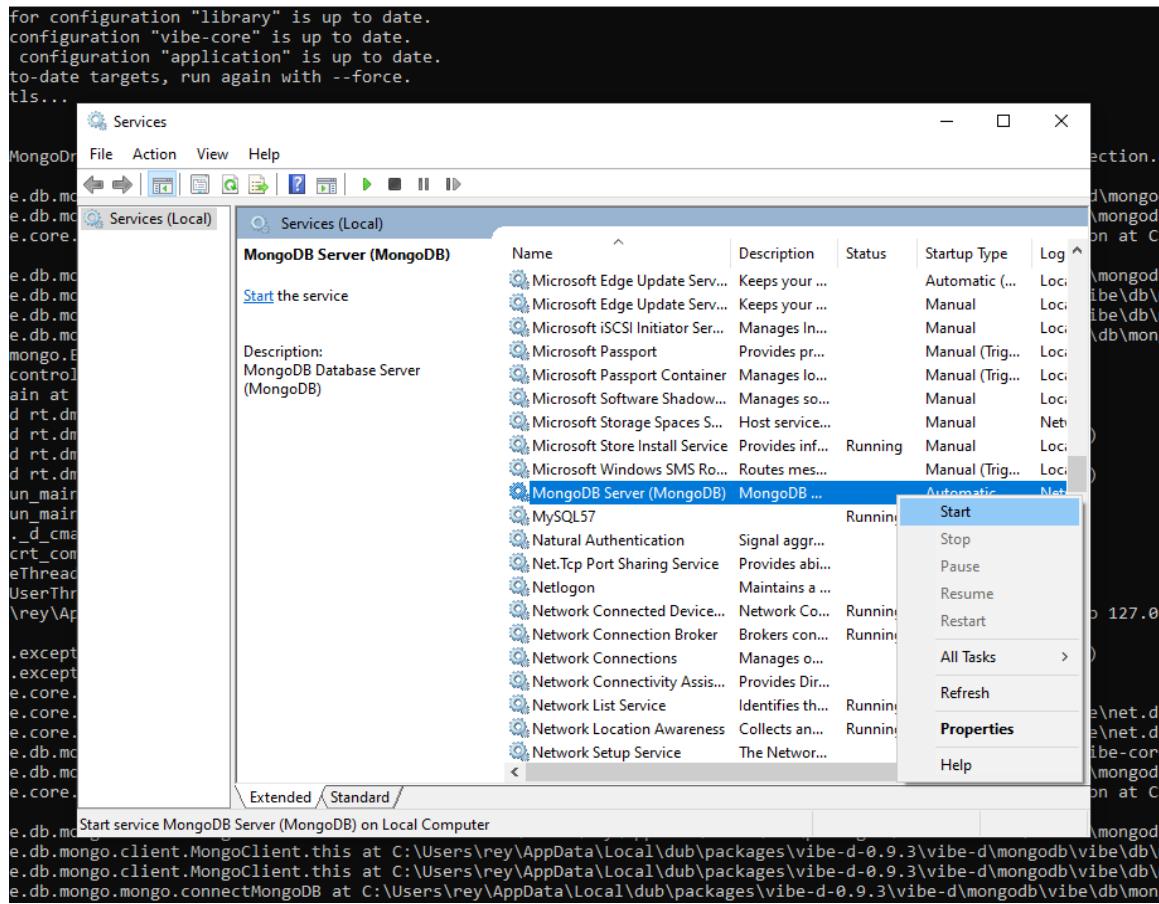
0x00007FF7DBA32F24      in      vibe.db.mongo.connectMongoDB      at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\mongodb\vibe\db\mongo\mongo.d(91)
0x00007FF7DBA26A81      in      empmongo.EmployeeModel.this      at
C:\vibeprojects\lorem\source\empmongo.d(34)
0x00007FF7DBA24F20      in      empcontrol.EmployeeController.this      at
C:\vibeprojects\lorem\source\empcontrol.d(45)
0x00007FF7DB9D3D56 in D main at C:\vibeprojects\lorem\source\app.d(8)
0x00007FF7DBED8D33 in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][])*).runAll().__lambda1()
0x00007FF7DBED8B4F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][])*).tryExec(scope void delegate())
0x00007FF7DBED8C5B in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][])*).runAll()
0x00007FF7DBED8B4F in void rt.dmain2._d_run_main2(char[][][], ulong, extern (C) int
function(char[][][])*).tryExec(scope void delegate())
0x00007FF7DBED8956 in d_run_main2
0x00007FF7DBEA2783 in d_run_main
0x00007FF7DB9D65C2      in      app._d_cmain!().main      at
C:\D\dmdd2\windows\bin\..\..\src\druntime\import\core\internal\entrypoint.d(29)
0x00007FF7DBFBAA558      in      __scrt_common_main_seh      at
d:\agent\_work\63\s\src\vctools\crt\vcstartup\src\startup\exe_common.inl(288)
0x00007FFF8FB67034 in BaseThreadInitThunk
0x00007FFFC01BD241 in RtlUserThreadStart
Program exited with code 1

```

C:\vibeprojects\lorem>

In this case, simply open the Services app of Windows and start MongoDB.





After starting the MongoDB server, try compiling and running again.

```
C:\vibeprojects\lorem>dub
Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.
taggedalgebraic 0.11.19: target for configuration "library" is up to date.
eventcore 0.9.13: target for configuration "winapi" is up to date.
stdx-allocator 2.77.5: target for configuration "library" is up to date.
vibe-core 1.13.0: target for configuration "winapi" is up to date.
vibe-d:utils 0.9.3: target for configuration "library" is up to date.
vibe-d:data 0.9.3: target for configuration "library" is up to date.
mir-linux-kernel 1.0.1: target for configuration "library" is up to date.
vibe-d:crypto 0.9.3: target for configuration "library" is up to date.
diet-ng 1.7.4: target for configuration "library" is up to date.
vibe-d:stream 0.9.3: target for configuration "library" is up to date.
vibe-d:textfilter 0.9.3: target for configuration "library" is up to date.
vibe-d:inet 0.9.3: target for configuration "library" is up to date.
vibe-d:tls 0.9.3: target for configuration "openssl-mscoff" is up to date.
vibe-d:http 0.9.3: target for configuration "library" is up to date.
vibe-d:mail 0.9.3: target for configuration "library" is up to date.
```

```
vibe-d:mongodb 0.9.3: target for configuration "library" is up to date.  
vibe-d:redis 0.9.3: target for configuration "library" is up to date.  
vibe-d:web 0.9.3: target for configuration "library" is up to date.  
vibe-d 0.9.3: target for configuration "vibe-core" is up to date.  
lorem ~master: target for configuration "application" is up to date.  
To force a rebuild of up-to-date targets, run again with --force.  
Copying files for vibe-d:tls...  
Running .\lorem.exe  
[main(----) INF] Listening for requests on http://[::1]:8080/  
[main(----) INF] Listening for requests on http://127.0.0.1:8080/
```

If you can refresh the browser without errors then you did good..

The EmployeeModel class

A web data-entry form in Vibe.d needs a structure from which we can extract the fields for saving into the database table. It makes things simpler if the database and the form have a one-to-one correspondence in field names.

We can create a struct data structure that mimics an employees record. We will call that struct Employee.

Then we can create a class that contains the methods to manipulate that struct, such as add, edit, get and delete, in short, all the CRUD operations (create, read, update, delete) on the MongoDB server. It is a model that represents one record of the employees table on the MongoDB server, so we will call that class EmployeeModel.

We are going to presume that this app will only be deployed in an intranet setting and only a few people have access, and that there is a rare chance of more than one person using the app at the same time.

Let us do that in a new project. Create a new project named lorem.

```
C:\vibeprojects>dub init lorem -t vibe.d
```

Then copy the files inside empapp\views\ folder of the previous project to save us time.

Create source\empmongo.d.

```
module empmongo;

import vibe.db.mongo.mongo;

struct Employee
{
    BsonObjectID _id;
    string dept;
    string email;
    string pword;
    string fname;
    string lname;
    string phone;
    string paygd;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}
```

```

struct Admin
{
    string email;
    string password;
}

class EmployeeModel
{
    MongoCollection emptable, admintable;

    this()
    {
        auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
        emptable = empdb["employees"];
        admintable = empdb["admins"];
    }

    void addEmployee(Employee e)
    {
        emptable.insert
        ([
            "deprt": e.deprt,
            "email": e.email,
            "pword": e.pword,
            "fname": e.fname,
            "lname": e.lname,
            "phone": e.phone,
            "paygd": e.paygd,
            "photo": e.photo,
            "street": e.street,
            "city": e.city,
            "province": e.province,
            "postcode": e.postcode
        ]);
    }
}

```

In the `this()` constructor of the `EmployeeModel` class, we get a connection to the MongoDB server running at localhost (127.0.0.1), then get a connection to the database (`empdb`) through it, then access the collections `employees` and `admins` through `empdb` and assign them to the `MongoCollection` variables `emptable` and `admintable`.

This is how to insert a record in MongoDB in Vibe.d syntax. The native MongoDB syntax is very, very similar.

We need to add new records to the blank `employees` collection so we should create an employee data-entry web form. Let's do that next.

The form for adding a new employee record

Edit views\menu.dt to add the New employee link.

```
div.menu-item
  a.item-link(href="/") Home
div.menu-item
  a.item-link(href="#find") Find employee
div.menu-item
  a.item-link(href="list_employees") Show employees
div.menu-item
  a.item-link(href="new_employee") New employee
div#find.modal-form
  div.modal-form-wrapper-find
    form.modal-form-grid(method="post", action="show_employee")
      input.text-
control(name="fname", type="text", placeholder=" First name", required)
      input.text-
control(name="lname", type="text", placeholder=" Last name", required)
      input#but-reset(type="reset", value="Clear")
      input#but-submit(type="submit", value="Submit")
  a.close(href="#close") Cancel
```

Then create the views\empnew.dt page for our data entry form.

```
extends layout
block maincontent
  include cssformgrid.dt
  div.form-grid-wrapper
    h2 New employee details
    form.form-grid(method="post", action="new_employee", enctype="multipart/form-data")
      div.form-grid-column Department<br />
        select#dept.form-grid-column-input(name="e_deprt")
          - foreach(dep; deps)
            option(value="#{dep}") #{dep}
      div.form-grid-column Salary grade<br />
        input.form-grid-column-
      input(type="text", name="e_paygd", placeholder="Salary grade")
      div.form-grid-column Email address<br />
        input.form-grid-column-
      input(type="email", name="e_email", placeholder="email@address.com", required)
      div.form-grid-column Password<br />
        input.form-grid-column-
      input(type="password", name="e_pword", placeholder="password", required)
      div.form-grid-column First name<br />
```

```

        input.form-grid-column-
input(type="text", name="e_fname", placeholder="First name", required)
    div.form-grid-column Last name<br />
        input.form-grid-column-
input(type="text", name="e_lname", placeholder="Last name", required)
    div.form-grid-column Phone<br />
        input.form-grid-column-
input(type="text", name="e_phone", placeholder="Phone number")
    div.form-grid-column Street address (without city)<br />
        input.form-grid-column-
input(type="text", name="e_street", placeholder="Street address", required)
    div.form-grid-column City<br />
        input.form-grid-column-
input(type="text", name="e_city", placeholder="City", required)
    div.form-grid-column Province<br />
        select#province.form-grid-column-input(name="e_province")
            - foreach(prov; provs)
                option(value="#{prov[0]}") #{prov[1]}
    div.form-grid-column Postal code<br />
        input.form-grid-column-
input(type="text", name="e_postcode", placeholder="A1A 1A1")
    div.form-grid-column ID Picture<br />
        input.form-grid-column-input(type="file", name="picture")
    input(type="hidden", name="e_photo")
    input(type="hidden", name="e_id", value="123456789012345678901234")
    div.form-grid-column
        br
        input.form-grid-column-button(type="reset", value="Clear")
    div.form-grid-column
        br
        input.form-grid-column-button(type="submit", value="Submit")

```

The line

```
form(method="post", action="new_employee", enctype="multipart/form-data")
```

will call the postNewEmployee() method (which we haven't created yet) of the EmployeeController class.

The part of the form declaration

```
enctype="multipart/form-data"
```

means the form will also upload a file, which in this case is a photo.

The form includes views\cssformgrid.dt for its CSS formatting.

Here is the views\cssformgrid.dt.

```
:css
.form-grid-wrapper
{
    width: 700px;
    height: auto;
    margin: 20px auto;
    padding: 5px 20px;
    border-radius: 10px;
    background-color: whitesmoke;
}
.form-grid
{
    display: grid;
    grid-template-columns: 1fr 1fr;
    gap: 5px;
}
.form-grid-column
{
    display: inline;
    height: auto;
    padding: 5px;
}
.form-grid-column-field
{
    width: 100%;
    font-size: 16px;
    border-bottom: 1px solid black;
    padding: 5px 0;
}
.form-grid-column-button
{
    width: 100%;
    font-size: 16px;
    height: 40px;
    margin-top: 10px;
}
.form-grid-column-input
{
    width: 100%;
    font-size: 16px;
}
```

The EmployeeController class

The EmployeeController class is the embodiment of the web framework as well as the controller bridging the model with the views.

Create source\empcontrol.d:

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;

class EmployeeController
{
    private EmployeeModel empModel;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ];
    string[][] provs =
    [
        ["AB", "Alberta"],
        ["BC", "British Columbia"],
        ["MB", "Manitoba"],
        ["NB", "New Brunswick"],
        ["NL", "Newfoundland and Labrador"],
        ["NS", "Nova Scotia"],
        ["NT", "Northwest Territories"],
        ["NU", "Nunavut"],
        ["ON", "Ontario"],
        ["PE", "Prince Edward Island"],
        ["QC", "Quebec"],
        ["SK", "Saskatchewan"],
        ["YT", "Yukon Territory"]
    ];
}
```

```

this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee()
{
    render!("empnew.dt", deps, provs);
}

void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic != null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
    redirect("list_employees");
}
}

```

We created the deps and the provs array variables for forms that require a drop-down options for departments and provinces.

The function

```
render!("index.dt");
```

is a variadic function template. It means `render!()` can have a varying number of parameters (arguments) of different data types. We have seen that construct before with `staticTemplate!()`.

```
render!("empnew.dt", deps, provs);
```

This time `render!()` has three arguments, one string and two arrays.

The file being uploaded is in `request.files`. Remember that the `HTTPServerRequest` and the `HTTPServerResponse` are available to us as the `request` and `response` variables.

So the line

```
auto pic = "picture" in request.files;
```

means we extracting the uploaded file from `request.files` into the `pic` variable.

In the `empnew.dt` form, the `picture` variable corresponds to the name we gave to the button field:

```
input.form-grid-column-input(type="file", name="picture")
```

The data-entry form requires a photo image file to be uploaded. Uploaded image files are usually not saved into the database as that would easily make the database bloated, so we need to save them into a folder which, if not already existing, we should create. Hence we need to import some libraries related to file operations:

```
import std.file;
import std.path;
import std.algorithm;
```

So we can call these file operation functions:

```
string ext = extension(pic.filename.name);
string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
if(canFind(exts, ext))
{
    photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
    string dir = "./public/uploads/photos/";
    mkdirRecurse(dir);
    string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
    try moveFile(pic.tempPath, NativePath(fullpath));
    catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
}
```

The uploaded photos will be saved in the \public\uploads\photos\ folder and the filename will be changed to the employee's name (firstname_lastname.ext).

The passwords need to be encrypted before they are saved so even if the database is hacked and bad guys gained access, the passwords are still safely unreadable. The employees may need to have access to their own data in the future, such as access to time worked and salary info, so we need to keep their passwords safe.

So we need to import the relevant library for encryption:

```
import vibe.http.auth.digest_auth;
```

so we can call this function:

```
e.pword = createDigestPassword(realm, e.email, e.pword);
```

We created the realm variable near the top:

```
string realm = "Lorem Ipsum Company";
```

because it is needed by the createDigestPassword() function. This function will be used by other methods too.

Vibe.d sometimes gives errors when importing data from MongoDB when there are missing fields because they might have been blank when the record was inserted, so we want to avoid blank fields. If some of the non-required fields are blank, we need to give them dummy data:

```
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
```

before saving the record to the database.

Now we can test the whole thing.

Test the whole thing

Let's check source\app.d.

```
import vibe.vibe;
import empcontrol;

void main()
{
    auto router = new URLRouter;
    router.get("*", serveStaticFiles("public/"));
    router.registerWebInterface(new EmployeeController);

    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];

    listenHTTP(settings, router);
    runApplication();
}
```

Then try to compile. If you get errors (the views\empnew.dt is a big file), they may be just simple typographical errors, like this:

```
500 - Internal Server Error

Internal Server Error

Internal error information:
object.Exception@C:\Users\rey\AppData\Local\dub\packages\vibe-d-
0.9.3\vibe-d\data\vibe\data\serialization.d(918): Missing non-optional
field 'id' of type 'Employee' (DefaultPolicy(T)).

-----
0x00007FF611495B23 in std.exception.bailOut!(object.Exception).bailOut
at C:\D\dmdd2\windows\bin\..\..\src\phobos\std\exception.d(516)
0x00007FF611495A29 in std.exception.enforce!().enforce!bool.enforce at
C:\D\dmdd2\windows\bin\..\..\src\phobos\std\exception.d(437)
0x00007FF6114FF20A in
vibe.data.serialization.deserializeValueImpl!(vibe.data bson.BsonSerial
izer,
DefaultPolicy).deserializeValueDeduced!(empmongo.Employee).deserializeV
alueDeduced at
C:\D\dmdd2\windows\bin\..\..\src\phobos\std\exception.d(434)
0x00007FF6114FF0FD in
vibe.data.serialization.deserializeValueImpl!(vibe.data bson.BsonSerial
izer,
DefaultPolicy).deserializeValue!(empmongo.Employee).deserializeValue at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\data\vibe\data\serialization.d(691)
0x00007FF6114FF0B5 in
vibe.data.serialization.deserializeWithPolicy!(vibe.data bson.BsonSerial
izer, DefaultPolicy, empmongo.Employee,
```

```
vibe.data.bson.Bson).deserializeWithPolicy at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\data\vibe\data\serialization.d(304)
0x00007FF6114FF031 in
vibe.data.serialization.deserialize!(vibe.data.bson.BsonSerializer,
empmongo.Employee, vibe.data.bson.Bson).deserialize at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\data\vibe\data\serialization.d(270)
0x00007FF6114FEE1 in
vibe.data.bson.deserializeBson!(empmongo.Employee).deserializeBson at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\data\vibe\data\bson.d(1217)
0x00007FF61150ADB2 in empmongo.EmployeeModel.getEmployees at
C:\vibeprojects\lorem\source\empmongo.d(80)
0x00007FF611507728 in loremService.LoremInterface.getListEmployees at
C:\vibeprojects\lorem\source\loremservice.d(166)
0x00007FF6114D7FFF in vibe.web.web.handleRequest!("getListEmployees",
getListEmployees, loremService.LoremInterface).handleRequest at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\web\vibe\web\web.d(1043)
0x00007FF61149FFDD in
vibe.web.web.registerWebInterface!(loremService.LoremInterface,
MethodStyle.lowerUnderscored).registerWebInterface.__lambda20 at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\web\vibe\web\web.d(214)
0x00007FF6115CB2F4 in
vibe.http.router.URLRouter.handleRequest.__lambda4 at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\router.d(218)
0x00007FF6115E294B in
vibe.http.router.MatchTree!(vibe.http.router.Route).MatchTree.doMatch
at C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\router.d(674)
0x00007FF6115E1C46 in
vibe.http.router.MatchTree!(vibe.http.router.Route).MatchTree.match at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\router.d(607)
0x00007FF6115CAC5D in vibe.http.router.URLRouter.handleRequest at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\router.d(211)
0x00007FF61164B50A in vibe.http.server.handleRequest at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(2292)
0x00007FF61160CD25 in vibe.http.server.handleHTTPConnection.__lambda4
at C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(253)
0x00007FF61160C587 in vibe.http.server.handleHTTPConnection at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(254)
0x00007FF6115E0279 in
vibe.http.server.listenHTTPPlain.doListen.__lambda6 at
C:\Users\rey\AppData\Local\dub\packages\vibe-d-0.9.3\vibe-
d\http\vibe\http\server.d(2048)
0x00007FF61188D255 in void vibe.core.task.TaskFuncInfo.set!(void
delegate(vibe.core.net.TCPConnection) @safe,
vibe.core.net.TCPConnection).set(ref void
delegate(vibe.core.net.TCPConnection) @safe, ref
```

```

vibe.core.net.TCPConnection).callDelegate(ref
vibe.core.task.TaskFuncInfo) at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\task.d-mixin-712(712)
0x00007FF611856596 in vibe.core.task.TaskFuncInfo.call at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\task.d(730)
0x00007FF611836AE6 in vibe.core.task.TaskFiber.run at
C:\Users\rey\AppData\Local\dub\packages\vibe-core-1.13.0\vibe-
core\source\vibe\core\task.d(439)
0x00007FF6119B64CF in void core.thread.context.Callable.opCall()
0x00007FF6119B8BA7 in fiber_entryPoint
0x00007FF61198C989 in pure nothrow @nogc void
core.thread.fiber.Fiber.initStack().trampoline()

```

It is a very long error message, but the culprit is just a typo in views\empnew.dt, which is e__id (double underscore) instead of id.

Once you resolved the errors, compile, run and refresh your browser.

Then click on New employee and enter some sample data to test the new data-entry system.

And click Submit, which will return this error:

404 - Not Found

Not Found

Internal error information:
No routes match path '/list_employees'

In the class EmployeeController, we have this line at the end of postNewEmployee() method:

```
void postNewEmployee(Employee e)
{
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string photopath = "none yet";
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
        e.photo = photopath;
    }
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
    redirect("list_employees");
}
```

The line

```
redirect("list_employees");
```

will become the URL http://localhost:8080/list_employees which corresponds to the method getListEmployees(). Which means it is passing control to the getListEmployees() method, which we haven't written yet.

Time to build the subsystem for listing the employees.

Showing all the employees

Edit source\empcontrol.d and add this code at the end:

```
void getListEmployees()
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}
```

Now, that's a very simple method. Two lines!

But it is calling the empModel.getEmployees() method, which we haven't written yet, and also needs the emplist.dt file, which we haven't created yet, so let's resolve that.

First, define the getEmployees() method in source\empmongo.d:

```
Employee[] getEmployees()
{
    Employee[] emps;
    auto results = emptable.find();
    foreach(doc;results) emps ~= deserializeBson!Employee(doc);
    return emps;
}
```

This method returns an Employee array, meaning, not just one record but all the records in the emptable collection. MongoDB saves documents (records) as Bson (binary JSON) objects, so we deserialize them using the Employee struct as the map.

If we use emptable.find() without any arguments, that means we want all the records.

Now let's create views\emplist.dt.

```
extends layout
block maincontent
    include csstable.dt
    div.table-wrapper
        table
            tr
                td.no-border(colspan="11")
            tr
                th Department
                th First name
                th Last name
                th Phone number
                th Email address
                th Salary grade
```

```

th Street address
th City
th Province
th PostCode
th Action
- foreach(e; emps)
tr
td #{e.deprt}
td #{e.fname}
td #{e.lname}
td #{e.phone}
td #{e.email}
td #{e.paygd}
td #{e.street}
td #{e.city}
td #{e.province}
td #{e.postcode}
td &nbsp;
form.form-hidden(method="get", action="edit_employee")
input(type="hidden", name="id", value="#{e._id}")
input(type="image", src="images/pen.ico")
| &nbsp;
form.form-hidden(method="get", action="delete_employee")
input(type="hidden", name="id", value="#{e._id}")
input(type="image", src="images/trash.ico")
| &nbsp;

```

We can mix D code inside a Diet template file with – (hyphen)

```
- foreach(e; emps)
```

Remember that we passed an Employee array emps to the template views\emplist.dt:

```

void getListEmployees()
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}

```

hence we can inject this code:

```
- foreach(e; emps)
```

which will iterate through all of emps, which is an array of Employees, with e representing one Employee record.

To display the value of a variable passed to the template, we use the construct `#{variable}`. This is equivalent to JSP's `<%= variable %>`

Hence the line

```
td #{e.deprt}
```

means display the value of `e.deprt`.

And here is `views\csstable.dt` which is needed by `emplist.dt`.

```
:css
.table-wrapper
{
    margin: 20px auto;
    padding: 20px;
}
table
{
    margin: 0 auto;
    padding: 20px;
    border-collapse: collapse;
}
table td, table th
{
    border: 1px solid black;
    margin: 0;
    padding: 0 5px;
}
.no-border
{
    border: 0;
}
```

Compile and run the app and refresh the browser. Click on New employee again and add an employee.

30/new_employee

w employees New employee

New employee details

Department Maintenance	Salary grade SG-1002
Email address jen@law.com	Password *****
First name Jen	Last name Law
Phone 12345678909	Street address (without city) 123 front st
City Vancouver	Province British Columbia
Postal code V5A 2G4	ID Picture <input type="button" value="Choose File"/> jennlaw.jpg
<input type="button" value="Clear"/> <input type="button" value="Submit"/>	

Then click the Submit button.

And you get this:

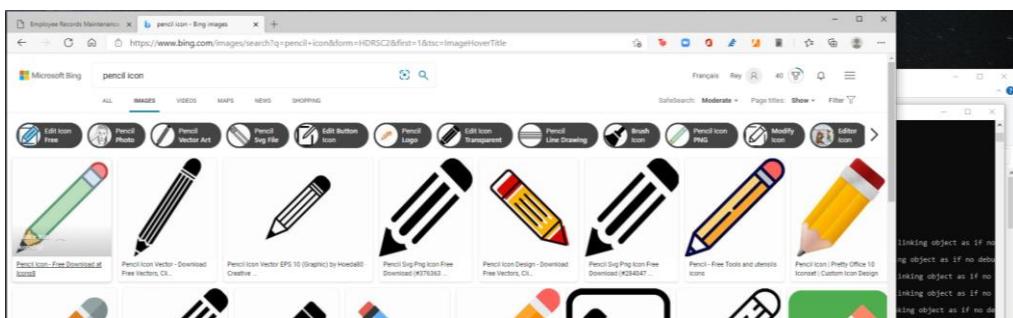
on - Bing images

0/list_employees

w employees New employee

Department	First name	Last name	Phone number	Email address	Salary grade	Street address	City	Province	PostCode	Action
Maintenance	Jen	Law	12345678909	jen@law.com	SG-1002	123 front st	Vancouver	BC	V5A 2G4	 

I got the icons for the pencil and the trash bin by simply googling for ‘pencil icon’ and ‘trash bin icon’ and saving the icons into the \public\images\ folder.



In the emplist.dt file, we have this lines

```
    input(type="image", src="images/pen.ico")
```

```
    input(type="image", src="images/trash.ico")
```

so save and name the icon files in the \public\images\ folder with the same names.

Add more sample records by clicking on the New employee link after each submission.

Here is the complete empmongo.d file:

```
module empmongo;

import vibe.db.mongo.mongo;

struct Employee
{
    BsonObjectID _id;
    string deprt;
    string email;
    string pword;
    string fname;
    string lname;
    string phone;
    string paygd;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}

struct Admin
{
    string email;
    string password;
}

class EmployeeModel
{
    MongoCollection emptable, admintable;

    this()
    {
        auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
        emptable = empdb["employees"];
        admintable = empdb["admins"];
    }
}
```

```

void addEmployee(Employee e)
{
    emptable.insert
    ([
        "deprt": e.deprt,
        "email": e.email,
        "pword": e.pword,
        "fname": e.fname,
        "lname": e.lname,
        "phone": e.phone,
        "paygd": e.paygd,
        "photo": e.photo,
        "street": e.street,
        "city": e.city,
        "province": e.province,
        "postcode": e.postcode
    ]);
}

Employee[] getEmployees()
{
    Employee[] emps;
    auto results = emptable.find();
    foreach(doc;results) emps ~= deserializeBson!Employee(doc);
    return emps;
}
}

```

And here is the full empcontrol.d file so far:

```

module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;

class EmployeeController
{
    private EmployeeModel empModel;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
    ]
}

```

```

    "Shipping and Receiving",
    "Purchasing and Supplies",
    "IT Services",
    "Human Resources",
    "Marketing"
];
string[][] provs =
[
    ["AB", "Alberta"],
    ["BC", "British Columbia"],
    ["MB", "Manitoba"],
    ["NB", "New Brunswick"],
    ["NL", "Newfoundland and Labrador"],
    ["NS", "Nova Scotia"],
    ["NT", "Northwest Territories"],
    ["NU", "Nunavut"],
    ["ON", "Ontario"],
    ["PE", "Prince Edward Island"],
    ["QC", "Quebec"],
    ["SK", "Saskatchewan"],
    ["YT", "Yukon Territory"]
];
this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee()
{
    render!("empnew.dt", deps, provs);
}

void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";

```

```

    mkdirRecurse(dir);
    string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
    try moveFile(pic.tempPath, NativePath(fullpath));
    catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
}
}

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.addEmployee(e);
redirect("list_employees");
}

void getListEmployees()
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}
}

```

Time to implement the editing of records to activate the pencil icon.

Retrieving and displaying an employee record for editing

First, we need to retrieve the record to be edited. Each record in a MongoDB collection has an `_id` field that has a unique value to identify the record.

There is a hidden field in the `emplist.dt` file: the `id` field, which has the value of the `_id` field in the database table, and which we can use as a key to retrieve the record so we can populate the form with the employee data.

```
form.form-hidden(method="get", action="edit_employee")
    input(type="hidden", name="id", value="#{e._id}")
    input(type="image", src="images/pen.ico")
```

Edit source\empmongo.d and add this method at the end:

```
Employee getEmployee(BsonObjectID id)
{
    Employee emp;
    auto result = emptable.findOne(["_id":id]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}
```

`BsonObjectID` is the Vibe.d data type of the `_id` field of the database. This method returns an `Employee` struct after retrieving the record from the database using the `_id` as the key.

Edit source\empcontrol.d and add this method at the end:

```
void getEditEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs);
}
```

This method calls the `empModel.getEmployee()` method with the `id` as the argument, then receives the returned `Employee` with the `_id` field equal to the `id` variable data and uses it to render the form `empedit.dt`, which we should create next.

Create views\empedit.dt.

```
extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Edit employee details
        form.form-grid(method="post", action="edit_employee", enctype="multipart/form-data")
```

```



```

```

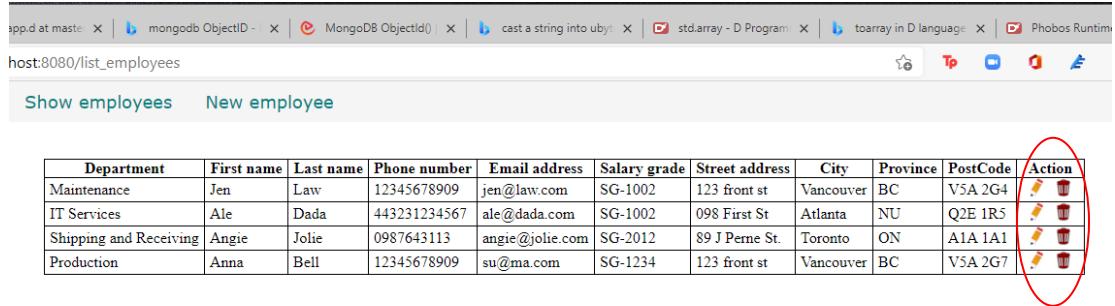

  



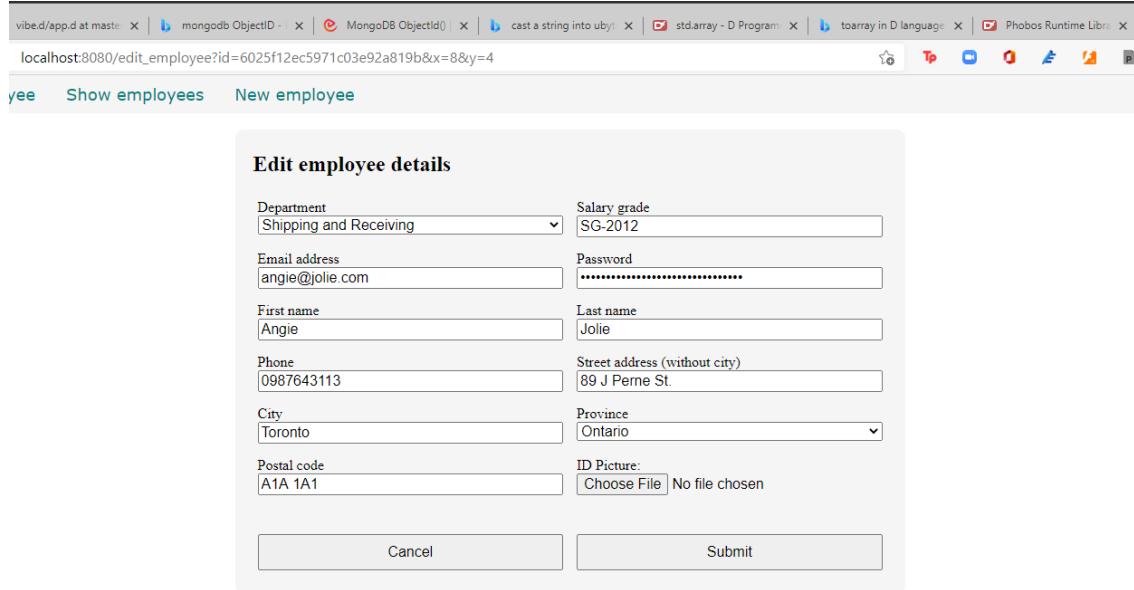
```

The form is displayed using the Employee data that was passed to it.

Compile, run and refresh the browser and click on a pencil icon to edit a record.



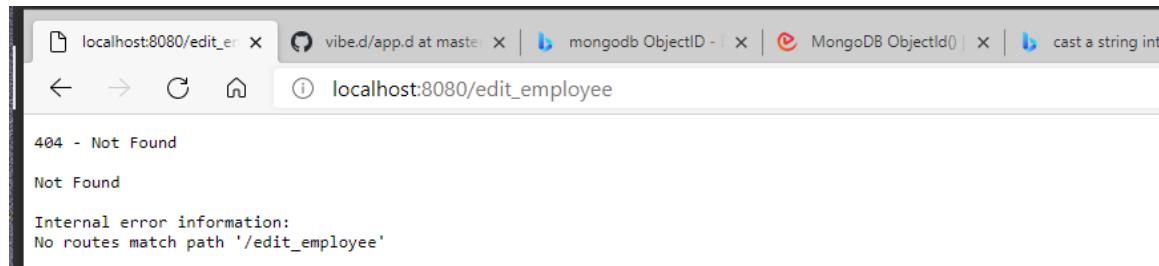
Department	First name	Last name	Phone number	Email address	Salary grade	Street address	City	Province	PostCode	Action
Maintenance	Jen	Law	12345678909	jen@law.com	SG-1002	123 front st	Vancouver	BC	V5A 2G4	 
IT Services	Ale	Dada	443231234567	ale@dada.com	SG-1002	098 First St	Atlanta	NU	Q2E 1R5	 
Shipping and Receiving	Angie	Jolie	0987643113	angie@jolie.com	SG-2012	89 J Perne St.	Toronto	ON	A1A 1A1	 
Production	Anna	Bell	12345678909	su@ma.com	SG-1234	123 front st	Vancouver	BC	V5A 2G7	 



Edit employee details

Department	Salary grade
Shipping and Receiving	SG-2012
Email address	Password
angie@jolie.com	*****
First name	Last name
Angie	Jolie
Phone	Street address (without city)
0987643113	89 J Perne St.
City	Province
Toronto	Ontario
Postal code	ID Picture:
A1A 1A1	<input type="button" value="Choose File"/> No file chosen
<input type="button" value="Cancel"/> <input type="button" value="Submit"/>	

Great, a record was opened for editing. After clicking submit, we get this error:



It means we haven't defined the functions to save the changes to the database. So let's do it.

Saving the form changes to the database

Add this method at the end of source\empcontrol.d.

```
void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}
```

The only differences between this method and the postNewEmployee() method are the first line and the second to the last line, which is the call to empModel.editEmployee() method, which we haven't defined yet.

Let's define the method at the end of source\empmodel.d.

```
void editEmployee(Employee e)
{
    emptable.update
    (
        ["_id":e._id],
        ["$set":
            [
                "deprt": e.deprt,
                "email": e.email,
                "pword": e.pword,
                "fname": e.fname,
```

```

        "lname": e.lname,
        "phone": e.phone,
        "paygd": e.paygd,
        "photo": e.photo,
        "street": e.street,
        "city": e.city,
        "province": e.province,
        "postcode": e.postcode
    ]
]
);
}

```

This code means look for the record with the value of the variable id in the _id field, then edit that record with this data.

Compile, run and refresh the browser and edit some files.

Department	First name	Last name	Phone number	Email address	Salary grade	Street address	City	Province	PostCode	Action
Maintenance	Jen	Law	12345678909	jen@law.com	SG-1002	123 front st	Vancouver	BC	V5A 2G4	
IT Services	Ale	Dada	443231234567	ale@dada.com	SG-1002	098 First St	Atlanta	NU	Q2E 1R5	
Shipping and Receiving	Angie	Jolie	0987643113	angie@jolie.com	SG-2012	89 J Perne St.	Toronto	ON	A1A 1A1	
Production	Anna	Bell	12345678909	su@ma.com	SG-1234	123 front st	Vancouver	BC	V5A 2G7	

Click on a pencil icon again to display that record for editing.

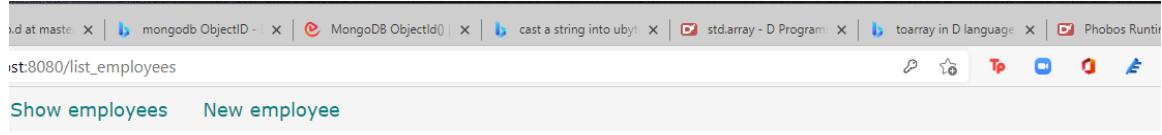
http://127.0.0.1:8080/list_employees

how employees New employee

Edit employee details

Department Shipping and Receiving	Salary grade SG-2012
Email address angie@jolie.com	Password *****
First name Angie	Last name Jolie
Phone 0987643113	Street address (without city) 89 J Perne St.
City Toronto	Province Ontario
Postal code A1A 1A1	ID Picture: Choose File No file chosen
<input type="button" value="Cancel"/> <input type="button" value="Submit"/>	

Make some changes to the record and press Submit. You should be able to see your changes saved.



Department	First name	Last name	Phone number	Email address	Salary grade	Street address	City	Province	PostCode	Action
Maintenance	Jen	Law	12345678909	jen@law.com	SG-1002	123 front st	Vancouver	BC	V5A 2G4	
IT Services	Ale	Dada	443231234567	ale@dada.com	SG-1002	098 First St	Atlanta	NU	Q2E 1R5	
Shipping and Receiving	Angie	Jolie	12345678909	angie@jolie.com	SG-2012	89 J Perre St.	Toronto	ON	A1A 1A1	
Production	Anna	Bell	12345678909	su@ma.com	SG-1234	123 front st	Vancouver	BC	V5A 2G7	

Here is the full source\empcontrol.d so far.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmysql;

class EmployeeController
{
    private EmployeeModel empModel;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ];
    string[][] provs =
    [
        ["AB", "Alberta"],
        ["BC", "British Columbia"],
        ["MB", "Manitoba"],
        ["NB", "New Brunswick"],
        ["NL", "Newfoundland and Labrador"],
        ["NS", "Nova Scotia"]
    ];
}
```

```

        [ "NT", "Northwest Territories"],
        [ "NU", "Nunavut"],
        [ "ON", "Ontario"],
        [ "PE", "Prince Edward Island"],
        [ "QC", "Quebec"],
        [ "SK", "Saskatchewan"],
        [ "YT", "Yukon Territory"]
    ];

    this()
    {
        empModel = new EmployeeModel;
    }

    void index()
    {
        render!("index.dt");
    }

    void getNewEmployee()
    {
        render!("empnew.dt", deps, provs);
    }

    void postNewEmployee(Employee e)
    {
        string photopath = "No photo submitted";
        auto pic = "picture" in request.files;
        if(pic !is null)
        {
            string ext = extension(pic.filename.name);
            string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
            if(canFind(exts, ext))
            {
                photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
                string dir = "./public/uploads/photos/";
                mkdirRecurse(dir);
                string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
                try moveFile(pic.tempPath, NativePath(fullpath));
                catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
            }
        }
        e.photo = photopath;
        if(e.phone.length == 0) e.phone = "(123) 456 7890";
        if(e.paygd.length == 0) e.paygd = "none yet";
        if(e.postcode.length == 0) e.postcode = "A1A 1A1";
        e.pword = createDigestPassword(realm, e.email, e.pword);
        empModel.addEmployee(e);
        redirect("list_employees");
    }
}

```

```

void getListEmployees()
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}

void getEditEmployee(int id)
{
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs);
}

void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}
}

```

And here is the full source\empmongo.d so far.

```

module empmongo;

import vibe.db.mongo.mongo;

struct Employee
{

```

```

BsonObjectID _id;
string dept;
string email;
string pword;
string fname;
string lname;
string phone;
string paygd;
string photo;
string street;
string city;
string province;
string postcode;
}

struct Admin
{
    string email;
    string password;
}

class EmployeeModel
{
    MongoCollection emptable, admintable;

    this()
    {
        auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
        emptable = empdb["employees"];
        admintable = empdb["admins"];
    }

    void addEmployee(Employee e)
    {
        emptable.insert
        ([
            "dept": e.dept,
            "email": e.email,
            "pword": e.pword,
            "fname": e.fname,
            "lname": e.lname,
            "phone": e.phone,
            "paygd": e.paygd,
            "photo": e.photo,
            "street": e.street,
            "city": e.city,
            "province": e.province,
            "postcode": e.postcode
        ]);
    }
}

```

```

Employee[] getEmployees()
{
    Employee[] emps;
    auto results = emptable.find();
    foreach(doc;results) emps ~= deserializeBson!Employee(doc);
    return emps;
}

Employee getEmployee(BsonObjectID id)
{
    Employee emp;
    auto result = emptable.findOne(["_id":id]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}

void editEmployee(Employee e)
{
    emptable.update
    (
        ["_id":e._id],
        ["$set":
            [
                "dept": e.deprt,
                "email": e.email,
                "pword": e.pword,
                "fname": e.fname,
                "lname": e.lname,
                "phone": e.phone,
                "paygd": e.paygd,
                "photo": e.photo,
                "street": e.street,
                "city": e.city,
                "province": e.province,
                "postcode": e.postcode
            ]
        ]
    );
}
}

```

Time to implement the empModel.delete() function to activate the trash bin icon.

Deleting records in the database

When a user clicks on the trash bin icon, it should not immediately delete the corresponding record. The user should be given the chance to recover if he/she made a mistake. We can simply display a page showing the employee details first, then ask the user for confirmation.

Edit source\empcontrol.d and append this code.

```
void getDeleteEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}
```

We retrieve the employee record and display that record in empdelete.dt.

Create views\empdelete.dt.

```
extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2.brown Are you sure you want to delete this record?
        form.form-grid(method="post", action="delete_employee")
            div.form-grid-column
                | Department:<br />
                div.form-grid-column-field #{e.deprt}
            div.form-grid-column
                | Salary grade:<br />
                div.form-grid-column-field #{e.paygd}
            div.form-grid-column
                | Email address:<br />
                div.form-grid-column-field #{e.email}
            div.form-grid-column
                | Password:<br />
                div.form-grid-column-field *****
            div.form-grid-column
                | First name:<br />
                div.form-grid-column-field #{e.fname}
            div.form-grid-column
                | Last name:<br />
                div.form-grid-column-field #{e.lname}
            div.form-grid-column
                | Phone:<br />
                div.form-grid-column-field #{e.phone}
            div.form-grid-column
                | Street address (without city):<br />
```

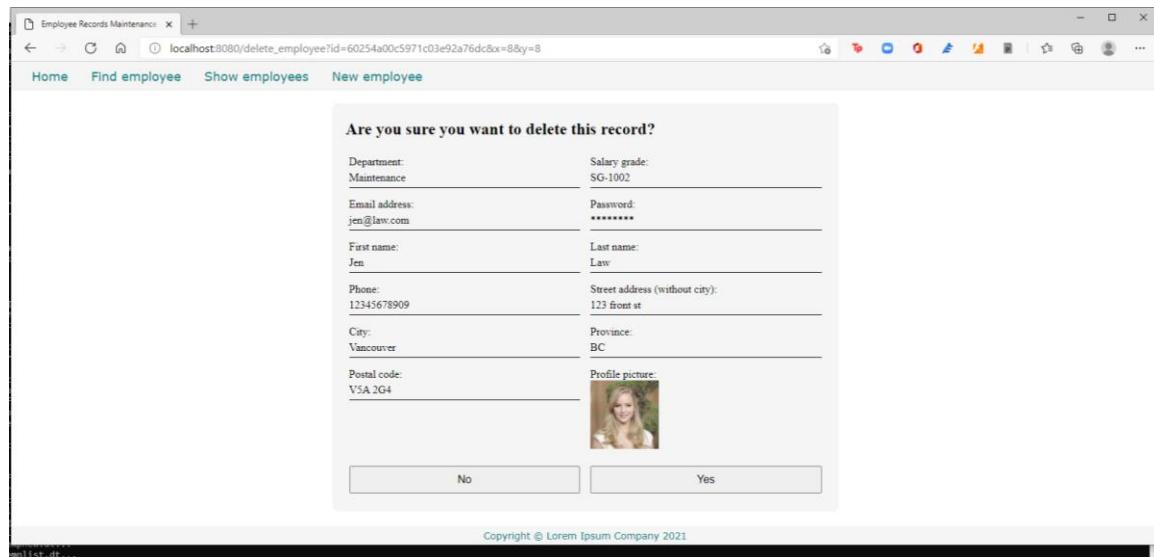
```



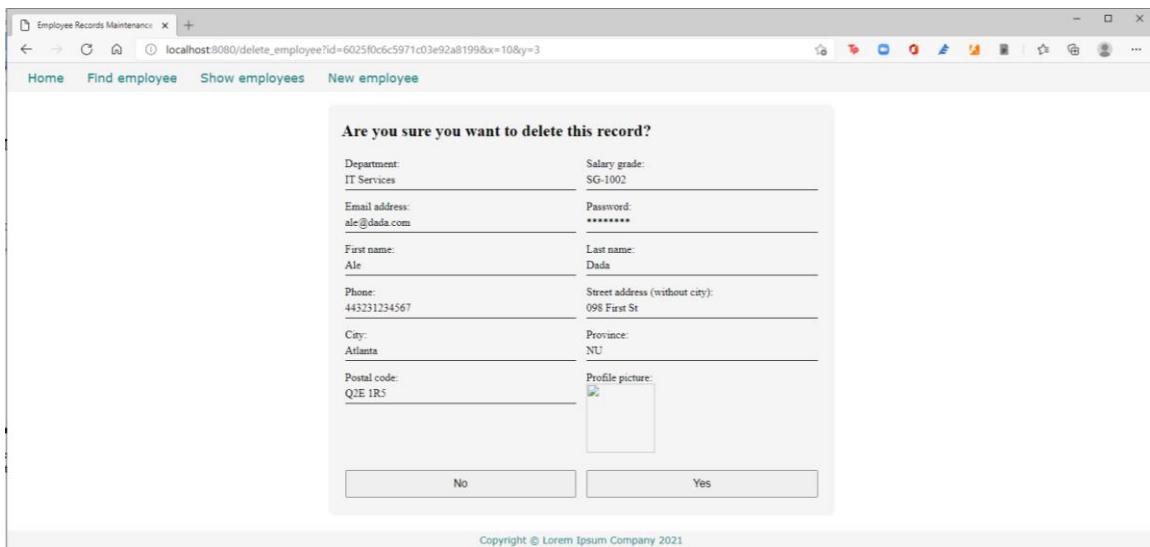
```

Here we displayed a page showing the employee details and asking the user for confirmation.

Compile, run and refresh the browser. Show the list of employees again and click on a trash bin icon. The confirmation page should show.

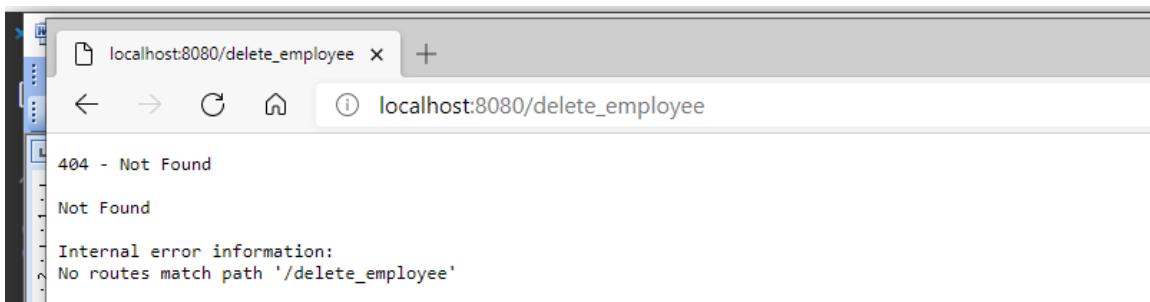


Click No for now just to test if the No button works. Then click the trash bin icon for another record.



This time click the Yes button.

You get this error.



So let's implement postDeleteEmployee() inside the EmployeeController class.

Edit source\empcontrol.d and append this code.

```
void postDeleteEmployee(BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}
```

We are calling empModel.deleteEmployee() here, so let's implement that.

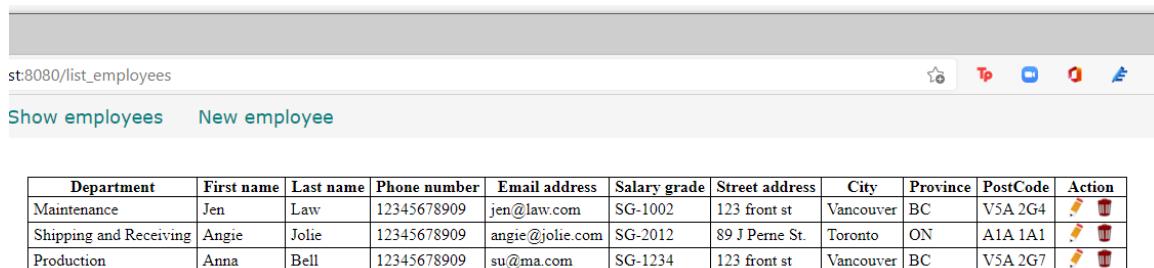
Edit source\empmongo.d and append this code.

```
void deleteEmployee(BsonObjectID id)
{
    emptable.remove([ "_id":id]);
}
```

Here we finally delete the record from the database table.

Compile, run and refresh the browser to the list of employees. Click on a trash bin icon, click Yes on the next screen, and you will be redirected to the list of employees again.

This time you should see that the record you selected is no longer listed.



Department	First name	Last name	Phone number	Email address	Salary grade	Street address	City	Province	PostCode	Action
Maintenance	Jen	Law	12345678909	jen@law.com	SG-1002	123 front st	Vancouver	BC	V5A 2G4	
Shipping and Receiving	Angie	Jolie	12345678909	angie@jolie.com	SG-2012	89 J Perne St.	Toronto	ON	A1A 1A1	
Production	Anna	Bell	12345678909	su@ma.com	SG-1234	123 front st	Vancouver	BC	V5A 2G7	

Here is the full source\empmongo.d so far.

```
module empmongo;

import vibe.db.mongo.mongo;

struct Employee
{
    BsonObjectID _id;
    string dept;
    string email;
    string pword;
    string fname;
    string lname;
    string phone;
    string paygd;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}

struct Admin
{
    string email;
    string password;
}

class EmployeeModel
{
    MongoCollection emptable, admintable;
```

```

this()
{
    auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
    emptable = empdb["employees"];
    admintable = empdb["admins"];
}

void addEmployee(Employee e)
{
    emptable.insert
    (
        [
            "deprt": e.deprt,
            "email": e.email,
            "pword": e.pword,
            "fname": e.fname,
            "lname": e.lname,
            "phone": e.phone,
            "paygd": e.paygd,
            "photo": e.photo,
            "street": e.street,
            "city": e.city,
            "province": e.province,
            "postcode": e.postcode
        ]);
}

Employee[] getEmployees()
{
    Employee[] emps;
    auto results = emptable.find();
    foreach(doc;results) emps ~= deserializeBson!Employee(doc);
    return emps;
}

Employee getEmployee(BsonObjectID id)
{
    Employee emp;
    auto result = emptable.findOne(["_id":id]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}

void editEmployee(Employee e)
{
    emptable.update
    (
        [ "_id":e._id],
        [ "$set":
            [

```

```

        "dept": e.deprt,
        "email": e.email,
        "pword": e.pword,
        "fname": e.fname,
        "lname": e.lname,
        "phone": e.phone,
        "paygd": e.paygd,
        "photo": e.photo,
        "street": e.street,
        "city": e.city,
        "province": e.province,
        "postcode": e.postcode
    ]
]
);
}

void deleteEmployee(BsonObjectID id)
{
    emptable.remove([ "_id":id]);
}
}

```

And here is the full source\empcontrol.d so far.

```

module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;

class EmployeeController
{
    private EmployeeModel empModel;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ]
}

```

```

];
string[][] provs =
[
    ["AB", "Alberta"],
    ["BC", "British Columbia"],
    ["MB", "Manitoba"],
    ["NB", "New Brunswick"],
    ["NL", "Newfoundland and Labrador"],
    ["NS", "Nova Scotia"],
    ["NT", "Northwest Territories"],
    ["NU", "Nunavut"],
    ["ON", "Ontario"],
    ["PE", "Prince Edward Island"],
    ["QC", "Quebec"],
    ["SK", "Saskatchewan"],
    ["YT", "Yukon Territory"]
];
this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee()
{
    render!("empnew.dt", deps, provs);
}

void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic != null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
}

```

```

    }

    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
    redirect("list_employees");
}

void getListEmployees()
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}

void getEditEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs);
}

void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}

```

```
void getDeleteEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

void postDeleteEmployee(BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}
```

Now, time to implement that Find employee link of the menu.

Finding and displaying an employee record

We have been finding and displaying an employee record already, so this should be trivial to us now. However, we have always used the `_id` field. This time, we are going to use the name of the employee as the key. After displaying the record, we should show some links about what the user should do with the record: edit it or simply dismiss the page and go back to the listing of employees.

Click on the Find employee menu item to test it.

The screenshot shows a web browser window titled "Employee Records Maintenance". The URL in the address bar is "localhost:8080/list_employees#find". The page contains a navigation bar with links: "Home", "Find employee", "Show employees", and "New employee". Below the navigation bar is a search form with two input fields: "First name" and "Last name", each with a placeholder value. There are also "Clear" and "Submit" buttons, and a "Cancel" link. To the right of the search form is a table displaying employee records. The table has columns: "First name", "Last name", "Phone number", "Email address", "Salary grade", and "Street". The data in the table is as follows:

	First name	Last name	Phone number	Email address	Salary grade	Street
Receiving	Jen	Law	12345678909	jen@law.com	SG-1002	123 f
Receiving	Angie	Jolie	12345678909	angie@jolie.com	SG-2012	89 J
	Anna	Bell	12345678909	su@ma.com	SG-1234	123 f

This time, we will find an employee by the first name and last name.

The screenshot shows the same web browser window and search interface as the previous one. However, the "First name" and "Last name" fields now contain the values "Anna" and "Bell" respectively. The rest of the page, including the table of employees, remains the same.

But when you click on submit, you get this error.

The screenshot shows a new browser tab titled "localhost:8080/show_employee". The URL in the address bar is "localhost:8080/show_employee". The page displays an error message: "404 - Not Found" and "Not Found". Below that, it says "Internal error information: No routes match path '/show_employee'".

It is looking for the getShowEmployee() function, so let's rectify that.

Edit source\empcontrol.d and append this code.

```
void getShowEmployee(string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    render!("empshow.dt", e);
}
```

We simply call an overloaded version of the EmployeeModel.getEmployee() function.

Edit source\empmongo.d and define this method at the end.

```
Employee getEmployee(string first, string last)
{
    Employee emp;
    auto result = emptable.findOne(["fname":first, "lname":last]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}
```

Instead of looking for the ID, this time the method is looking for the first name and last name.

Create views\empshow.dt.

```
extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Employee details
        form.form-grid(method="get", action="edit_employee")
            div.form-grid-column
                | Department<br />
                div.form-grid-column-field #{e.deprt}
            div.form-grid-column
                | Salary grade<br />
                div.form-grid-column-field #{e.paygd}
            div.form-grid-column
                | Email address<br />
                div.form-grid-column-field #{e.email}
            div.form-grid-column
                | Password<br />
                div.form-grid-column-field *****
            div.form-grid-column
                | First name<br />
                div.form-grid-column-field #{e.fname}
```

```



```

Then compile, run and refresh the browser. Look for an employee and click Submit.

	First name	Last name	Phone number
Receiving	Jen	Law	12345678909
	Angie	Jolie	12345678909
	Anna	Bell	12345678909

A vertical scrollbar is visible on the left side of the table area."/>

If the employee is not found, it simply shows the home page.

But if the employee is found, this will be displayed.

The screenshot shows a web browser window with the URL `localhost:8080/show_employee?fname=Jen&lname=Law`. The page title is "Employee details". The form fields are as follows:

Department Maintenance	Salary grade SG-1002
Email address jen@law.com	Password *****
First name Jen	Last name Law
Phone 12345678909	Street address (without city) 123 front st
City Vancouver	Province BC
Postal code V5A 2G4	Profile picture

At the bottom are two buttons: "Close" and "Edit".

Click on the Edit button. It will show the same empedit.dt file we created previously.

The screenshot shows a web browser window with the URL `localhost:8080/edit_employee?id=60254a00c5971c03e92a76dc&x=98y=3`. The page title is "Edit employee details". The form fields are identical to the previous screenshot, with the addition of a file input field for the profile picture:

Department Maintenance	Salary grade SG-1002
Email address jen@law.com	Password *****
First name Jen	Last name Law
Phone 12345678909	Street address (without city) 123 Front St
City Vancouver	Province British Columbia
Postal code V5A 2G4	ID Picture <input type="file"/> Choose File No file chosen

At the bottom are two buttons: "Cancel" and "Submit".

Here is the full code of source\empcontrol.d so far.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
```

```
import empmongo;

class EmployeeController
{
    private EmployeeModel empModel;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ];
    string[][][] provs =
    [
        ["AB", "Alberta"],
        ["BC", "British Columbia"],
        ["MB", "Manitoba"],
        ["NB", "New Brunswick"],
        ["NL", "Newfoundland and Labrador"],
        ["NS", "Nova Scotia"],
        ["NT", "Northwest Territories"],
        ["NU", "Nunavut"],
        ["ON", "Ontario"],
        ["PE", "Prince Edward Island"],
        ["QC", "Quebec"],
        ["SK", "Saskatchewan"],
        ["YT", "Yukon Territory"]
    ];
}

this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee()
{
    render!("empnew.dt", deps, provs);
}
```

```

void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
    redirect("list_employees");
}

void getListEmployees()
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}

void getEditEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs);
}

void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {

```

```

        photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullpath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
    }
}

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.editEmployee(e);
redirect("list_employees");
}

void getDeleteEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

void postDeleteEmployee(BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

void getShowEmployee(string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    render!("empshow.dt", e);
}
}

```

And here is the full code for the source\empmongo.d so far.

```

module empmongo;

import vibe.db.mongo.mongo;

struct Employee
{
    BsonObjectID _id;
    string dept;
    string email;
    string pword;
    string fname;
}

```

```
    string lname;
    string phone;
    string paygd;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}

struct Admin
{
    string email;
    string password;
}

class EmployeeModel
{
    MongoCollection emptable, admintable;

    this()
    {
        auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
        emptable = empdb["employees"];
        admintable = empdb["admins"];
    }

    void addEmployee(Employee e)
    {
        emptable.insert
        ([
            "deprt": e.deprt,
            "email": e.email,
            "pword": e.pword,
            "fname": e.fname,
            "lname": e.lname,
            "phone": e.phone,
            "paygd": e.paygd,
            "photo": e.photo,
            "street": e.street,
            "city": e.city,
            "province": e.province,
            "postcode": e.postcode
        ]);
    }

    Employee[] getEmployees()
    {
        Employee[] emps;
        auto results = emptable.find();
    }
}
```

```

        foreach(doc;results) emps ~= deserializeBson!Employee(doc);
        return emps;
    }

Employee getEmployee(BsonObjectID id)
{
    Employee emp;
    auto result = emptable.findOne(["_id":id]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}

void editEmployee(Employee e)
{
    emptable.update
    (
        ["_id":e._id],
        ["$set":
            [
                "deprt": e.deprt,
                "email": e.email,
                "pword": e.pword,
                "fname": e.fname,
                "lname": e.lname,
                "phone": e.phone,
                "paygd": e.paygd,
                "photo": e.photo,
                "street": e.street,
                "city": e.city,
                "province": e.province,
                "postcode": e.postcode
            ]
        ]
    );
}

void deleteEmployee(BsonObjectID id)
{
    emptable.remove(["_id":id]);
}

Employee getEmployee(string first, string last)
{
    Employee emp;
    auto result = emptable.findOne(["fname":first, "lname":last]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}

```

Capturing and displaying (some) error messages

When a new employee record failed to be added to the database because of some error, that error should be shown to the user to let the user know why the insertion failed so the user can try to resolve the problem.

When an update attempt failed, the error should also be shown to let the user know the state of things.

After a failed search when the employee was not found, the message should be also be displayed.

When a login attempt failed, the login page should display again with an error message telling the user why the login attempt failed.

When a method in EmployeeController generates an error, the error message is captured in the provided `_error` variable, which can be used to display the error. The facility to display this `_error` message is provided by the `@errorDisplay` annotation. The `@errorDisplay` annotation indicates which method will handle the error.

Edit the source\empcontrol.d to make use of this facility.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;

struct User
{
    bool loggedIn;
    string email;
}

class EmployeeController
{
    private EmployeeModel empModel;
    private SessionVar!(User, "user") m_user;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
    ]
}
```

```

    "Shipping and Receiving",
    "Purchasing and Supplies",
    "IT Services",
    "Human Resources",
    "Marketing"
];
string[][] provs =
[
    ["AB", "Alberta"],
    ["BC", "British Columbia"],
    ["MB", "Manitoba"],
    ["NB", "New Brunswick"],
    ["NL", "Newfoundland and Labrador"],
    ["NS", "Nova Scotia"],
    ["NT", "Northwest Territories"],
    ["NU", "Nunavut"],
    ["ON", "Ontario"],
    ["PE", "Prince Edward Island"],
    ["QC", "Quebec"],
    ["SK", "Saskatchewan"],
    ["YT", "Yukon Territory"]
];
this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee(string _error = null)
{
    string error = _error;
    render!("empnew.dt", deps, provs, error);
}

@errorDisplay!getNewEmployee
void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))

```

```

    {
        photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullpath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
    }
}

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.addEmployee(e);
redirect("list_employees");
}

void getListEmployees(string _error = null)
{
    string error = _error;
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps, error);
}

void getEditEmployee(BsonObjectID id, string _error = null)
{
    string error = _error;
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
}

```

```

    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}

void getDeleteEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

void postDeleteEmployee(BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
void getShowEmployee(string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    enforce(e != Employee.init, "Employee not found!");
    render!("empshow.dt", e);
}
}

```

Let us take the getEditEmployee() example.

```

void getEditEmployee(BsonObjectID id, string _error = null)
{
    Employee e = empModel.getEmployee(id);
    string error = _error;
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)

```

Because we indicated in postEditEmployee() here:

```
@errorDisplay!getEditEmployee  
void postEditEmployee(Employee e)
```

that getEditEmployee() will be called when there is an error during processing, the getEditEmployee() method now has an argument named _error.

```
void getEditEmployee(BsonObjectID id, string _error = null)
```

The _error argument is assigned null as a default in case no errors were generated. We can now pass it to the empedit.dt template.

```
string error = _error;  
render!("empedit.dt", e, deps, provs, error);
```

Edit views\empedit.dt to add the error message at the bottom.

```
extends layout  
block maincontent  
    include cssformgrid.dt  
    div.form-grid-wrapper  
        h2 Edit employee details  
        form.form-grid(method="post", action="edit_employee", enctype="multipart/form-data")  
            div.form-grid-column Department<br />  
                select#depid.form-grid-column-input(name="e_deprt", value="#{e.deprt}")  
                    - foreach(dep; deps)  
                        - if(dep == e.deprt)  
                            option(value="#{dep}", selected) #{dep}  
                        - else  
                            option(value="#{dep}") #{dep}  
            div.form-grid-column Salary grade<br />  
                input.form-grid-column-  
input(type="text", name="e_paygd", value="#{e.paygd}")  
            div.form-grid-column Email address<br />  
                input.form-grid-column-  
input(type="email", name="e_email", value="#{e.email}")  
            div.form-grid-column Password<br />  
                input.form-grid-column-  
input(type="password", name="e_pword", value="#{e.pword}")  
            div.form-grid-column First name<br />  
                input.form-grid-column-  
input(type="text", name="e_fname", value="#{e.fname}")  
            div.form-grid-column Last name<br />  
                input.form-grid-column-  
input(type="text", name="e_lname", value="#{e.lname}")  
            div.form-grid-column Phone<br />  
                input.form-grid-column-  
input(type="text", name="e_phone", value="#{e.phone}")  
            div.form-grid-column Street address (without city)<br />
```

```

        input.form-grid-column-
input(type="text", name="e_street", value="#{e.street}")
    div.form-grid-column City<br />
        input.form-grid-column-
input(type="text", name="e_city", value="#{e.city}")
    div.form-grid-column Province<br />
        select#province.form-grid-column-
input(name="e_province", value="#{e.province}")
    - foreach(p; provs)
        - if(p[0] == e.province)
            option(value="#{p[0]}", selected) #{p[1]}
        - else
            option(value="#{p[0]}") #{p[1]}
    div.form-grid-column Postal code<br />
        input.form-grid-column-
input(type="text", name="e_postcode", value="#{e.postcode}")
    div.form-grid-column ID Picture:<br />
        input.form-grid-column-input(type="file", name="picture")
input(type="hidden", name="e_photo", value="#{e.photo}")
input(type="hidden", name="e_id", value="#{e._id}")
div.form-grid-column
    br
    a(href="list_employees")
        button.form-grid-column-button(type="button") Cancel
div.form-grid-column
    br
    input.form-grid-column-button(type="submit", value="Submit")
- if(error)
    div.error #{error}

```

Edit views\emplist.dt to add the error message at the top.

```

extends layout
block maincontent
    include csstable.dt
    div.table-wrapper
        table
            tr
                td.no-border(colspan="11")
                    - if(error)
                        span.error #{error}<br /><br />
            tr
                th Department
                th First name
                th Last name
                th Phone number
                th Email address
                th Salary grade

```

```

th Street address
th City
th Province
th PostCode
th Action
- foreach(e; emps)
tr
td #{e.deprt}
td #{e.fname}
td #{e.lname}
td #{e.phone}
td #{e.email}
td #{e.paygd}
td #{e.street}
td #{e.city}
td #{e.province}
td #{e.postcode}
td &nbs;
form.form-hidden(method="get", action="edit_employee")
input(type="hidden", name="id", value="#{e._id}")
input(type="image", src="images/pen.ico")
| &nbs;
form.form-hidden(method="get", action="delete_employee")
input(type="hidden", name="id", value="#{e._id}")
input(type="image", src="images/trash.ico")
| &nbs;

```

Edit views\empnew.dt and display the error message at the bottom.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 New employee details
        form.form-grid(method="post", action="new_employee", enctype="multipart/form-data")
            div.form-grid-column Department<br />
                select#dept.form-grid-column-input(name="e_deprt")
                    - foreach(dep; deps)
                        option(value="#{dep}") #{dep}
            div.form-grid-column Salary grade<br />
                input.form-grid-column-
input(type="text", name="e_paygd", placeholder="Salary grade")
            div.form-grid-column Email address<br />
                input.form-grid-column-
input(type="email", name="e_email", placeholder="email@address.com", required)
            div.form-grid-column Password<br />
                input.form-grid-column-
input(type="password", name="e_pword", placeholder="password", required)

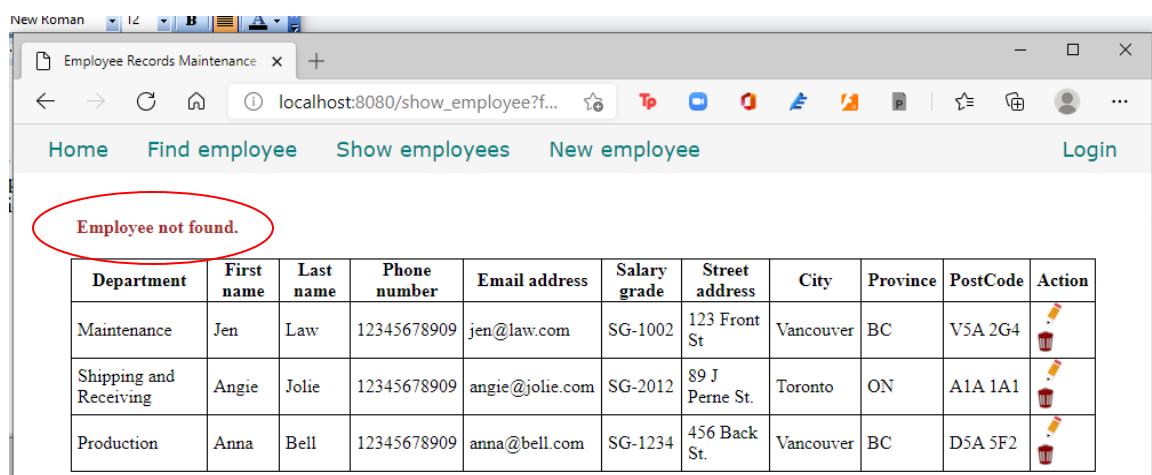
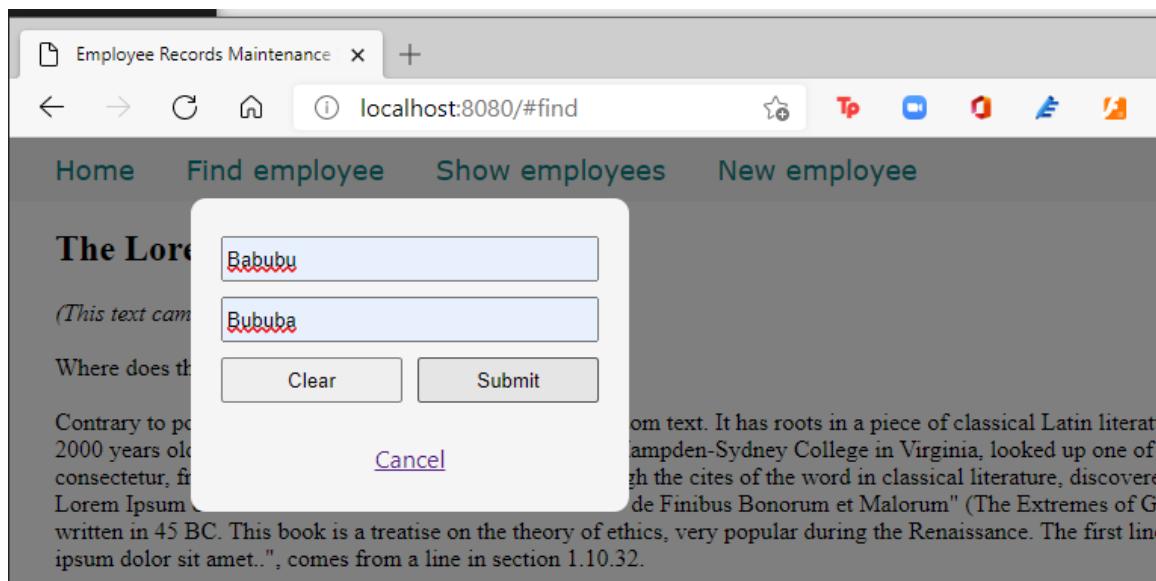
```

```



```

Compile, run and refresh the browser. Click on Find employee and enter a non-existent record and click Submit.



So now we can display error messages.

Here is the complete code for source\empcontrol.d so far.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmysql;

class EmployeeController
{
    private EmployeeModel empModel;
    private SessionVar!(User, "user") m_user;
```

```

string realm = "Lorem Ipsum Company";
string[] deps =
[
    "Management and Admin",
    "Accounting and Finance",
    "Production",
    "Maintenance",
    "Shipping and Receiving",
    "Purchasing and Supplies",
    "IT Services",
    "Human Resources",
    "Marketing"
];
string[][] provs =
[
    ["AB", "Alberta"],
    ["BC", "British Columbia"],
    ["MB", "Manitoba"],
    ["NB", "New Brunswick"],
    ["NL", "Newfoundland and Labrador"],
    ["NS", "Nova Scotia"],
    ["NT", "Northwest Territories"],
    ["NU", "Nunavut"],
    ["ON", "Ontario"],
    ["PE", "Prince Edward Island"],
    ["QC", "Quebec"],
    ["SK", "Saskatchewan"],
    ["YT", "Yukon Territory"]
];

this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee(string _error = null)
{
    string error = _error;
    render!("empnew.dt", deps, provs, error);
}

@errorDisplay!getNewEmployee
void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
}

```

```

auto pic = "picture" in request.files;
if(pic !is null)
{
    string ext = extension(pic.filename.name);
    string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
    if(canFind(exts, ext))
    {
        photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullpath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
    }
}
e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.addEmployee(e);
redirect("list_employees");
}

void getListEmployees(string _error = null)
{
    string error = _error;
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps, error);
}

void getEditEmployee(int id, string _error = null)
{
    string error = _error;
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.updateEmployee(e);
    redirect("list_employees");
}

```

```

        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullname = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullpath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
    }
}

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.editEmployee(e);
redirect("list_employees");
}

void getDeleteEmployee(int id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

void postDeleteEmployee(int id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
void getShowEmployee(string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    enforce(e != Employee.init, "Employee not found!");
    render!("empshow.dt", e);
}
}

```

Time to secure our site so that only authorized users can make changes to the data.

Time to talk about logging in, authentication and authorization.

Logging in, authentication and authorization

We need to restrict access to sensitive parts of our site so only authorized users (admins) can view and make changes to sensitive data. We need a login system to authenticate and authorize users.

Authentication means verifying that the user is who he/she says he/she is. Authorization means determining which parts of the app the user is allowed access.

In a commercial company, only a limited number of people are supposed to access sensitive data such as employee records.

We removed the Login menu item before. Time to put it back on the menu (pun intended), but this time it should point to a regular form instead of a modal form.

Edit views\menu.dt.

```
div.menu-item
  a.item-link(href="/") Home
div.menu-item
  a.item-link(href="#find") Find employee
div.menu-item
  a.item-link(href="list_employees") Show employees
div.menu-item
  a.item-link(href="new_employee") New employee
div.menu-item
  a.item-link.item-link-right(href="login") Login
div#find.modal-form
  div.modal-form-wrapper-find
    form.modal-form-grid(method="get", action="show_employee")
      input.text-
control(name="fname", type="text", placeholder=" First name", required)
  input.text-
control(name="lname", type="text", placeholder=" Last name", required)
  input#but-reset(type="reset", value="Clear")
  input#but-submit(type="submit", value="Submit")
  a.close(href="#close") Cancel
```

We simply add the Login link to display the login page.

Edit source\empcontrol.d and define the getLogin() method.

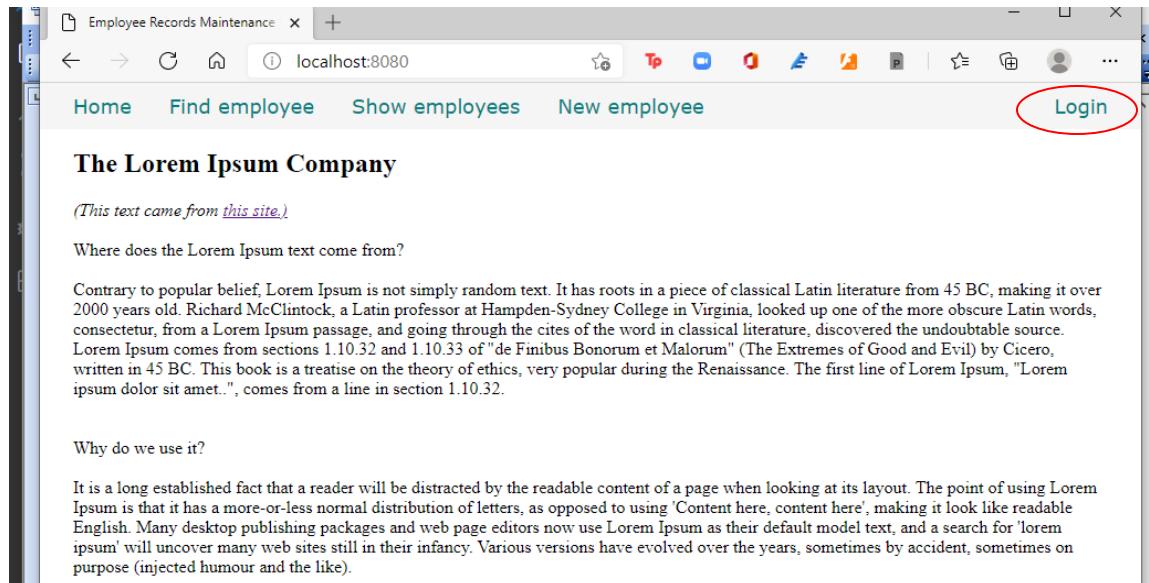
```
void getLogin()
{
  render!"login.dt";
}
```

The code is simply rendering the login.dt template file.

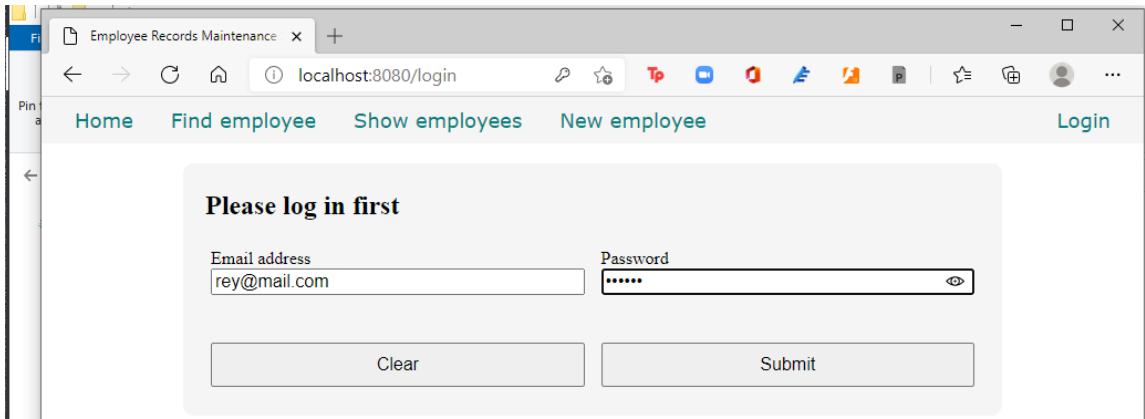
Let's create views\login.dt.

```
extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Please log in first
        form.form-grid(method="post", action="login")
            div.form-grid-column Email address<br />
                input.form-grid-column-
input(type="email", name="email", placeholder="email", required)
            div.form-grid-column Password<br />
                input.form-grid-column-
input(type="password", name="password", placeholder="password", required)
            div.form-grid-column
                br
                input.form-grid-column-button(type="reset", value="Clear")
            div.form-grid-column
                br
                input.form-grid-column-button(type="submit", value="Submit")
```

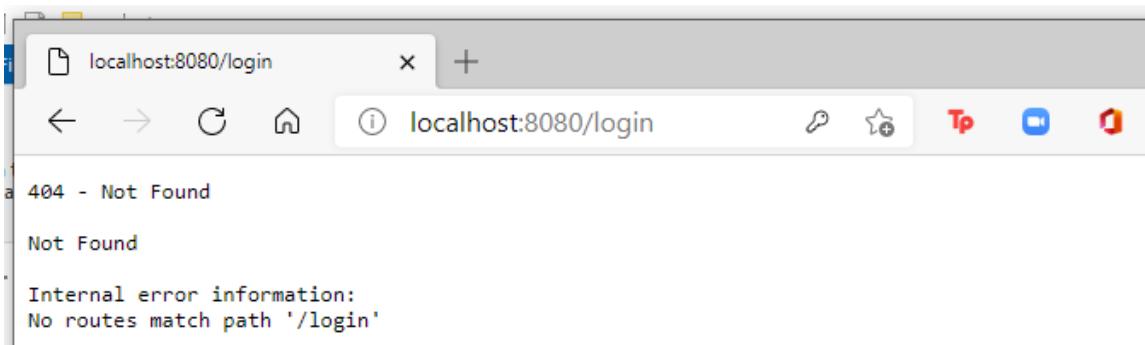
Compile, run and refresh the browser.



Click on the Login link on the menu to show the login page.



But when you try to login, you get this error:



Let us fix that. But first, let us encrypt the passwords in the admins collection.

Looking at the data in the admins collection, we see that the passwords remain raw and readable instead of encrypted so it needs to be rectified.

The screenshot shows the MongoDB Compass interface. On the left, the sidebar shows "HOST localhost:27017", "CLUSTER Standalone", and "EDITION MongoDB 4.4.3 Community". The "empdb" database is selected, and its "admins" collection is shown. The collection has 2 documents. The first document has fields: "_id: ObjectId("6027e8a7130af8c328d2c299")", "email: "admin1@lorem.com\"", and "password: "secret" (with the word "secret" circled in red). The second document has fields: "_id: ObjectId("6027e8a7130af8c328d2c29a")", "email: "admin2@lorem.com\"", and "password: "secret" (with the word "secret" circled in red).

_id	email	password
ObjectId("6027e8a7130af8c328d2c299")	admin1@lorem.com"	"secret"
ObjectId("6027e8a7130af8c328d2c29a")	admin2@lorem.com"	"secret"

Encrypting the passwords in the admins table

Edit views\menu.dt.

```
div.menu-item
  a.item-link(href="/") Home
div.menu-item
  a.item-link(href="#find") Find employee
div.menu-item
  a.item-link(href="list_employees") Show employees
div.menu-item
  a.item-link(href="encrypt_passwords") Encrypt passwords
div.menu-item
  a.item-link(href="new_employee") New employee
div.menu-item
  a.item-link.item-link-right(href="login") Login
div#find.modal-form
  div.modal-form-wrapper-find
    form.modal-form-grid(method="get", action="show_employee")
      input.text-
control(name="fname", type="text", placeholder=" First name", required)
  input.text-
control(name="lname", type="text", placeholder=" Last name", required)
  input#but-reset(type="reset", value="Clear")
  input#but-submit(type="submit", value="Submit")
  a.close(href="#close") Cancel
```

Edit source\empcontrol.d and add this code.

```
void getEncryptPasswords()
{
  Admin[] admins = empModel.getAdmins();
  foreach(a; admins)
  {
    auto pword = createDigestPassword(realm, a.email, a.password);
    empModel.replacePassword(a.email, pword);
  }
  redirect("/");
}
```

Edit source\empmongo.d and add this code.

```
Admin[] getAdmins()
{
  Admin[] admins;
  auto result = admintable.find();
  foreach(doc; result) admins ~= deserializeBson!Admin(doc);
  return admins;
```

```

}

void replacePassword(string email, string password)
{
    admintable.update(["email":email], ["$set": {"password":password}]);
}

```

There must be a more efficient way to update the same field in all records in one go, but since I am no MongoDB expert, I'll leave that to you.

Compile, run and refresh the browser, then click on the Encrypt passwords link.

The screenshot shows a browser window with several tabs open. The active tab is titled "Employee Records Maintenance". The menu bar includes "Home", "Find employee", "Show employees", "Encrypt passwords" (which is circled in red), "New employee", and "Login". Below the menu, the text "The Lorem Ipsum Company" is displayed, followed by a note about the text source and a question about its origin.

It looks as if nothing happened. Verify with Compass if the passwords were really encrypted by clicking on the REFRESH button.

The screenshot shows the MongoDB Compass interface connected to "localhost:27017/empdb.admins". The left sidebar shows "Local" selected, with "4 DBS" and "2 COLLECTIONS" listed. The main pane shows the "empdb.admins" collection with 2 documents. The "password" field values are circled in red. The "REFRESH" button at the bottom right of the document list is also circled in red.

_id	email	password
<code>ObjectId("6027e8a7130af8c328d2c299")</code>	"admin@lorem.com"	"b6296f83555e5120c03c5a2449ad557b"
<code>ObjectId("6027e8a7130af8c328d2c29a")</code>	"admin@lorem.com"	"4e548f23b65348b109efab9ef0b03fbc"

The passwords were really encrypted. Now we can remove the temporary link from the menu, but we can let the code for encrypting passwords of the admins table stay for future use.

Edit `views\menu.dt` to remove the link.

```


Home


```

```


.menu-item
  a.item-link(href="#find") Find employee


.menu-item
  a.item-link(href="list_employees") Show employees


.menu-item
  a.item-link(href="new_employee") New employee


.menu-item
  a.item-link.item-link-right(href="login") Login



Compile, run and refresh the browser to verify.



The link to encrypt the passwords is gone.



This is the state of source\empcontrol.d so far.



```

module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;

class EmployeeController
{
 private EmployeeModel empModel;
 string realm = "Lorem Ipsum Company";
 string[] deps =
 [
 "Management and Admin",
 "Accounting and Finance",

```


```

```

    "Production",
    "Maintenance",
    "Shipping and Receiving",
    "Purchasing and Supplies",
    "IT Services",
    "Human Resources",
    "Marketing"
];
string[][] provs =
[
    ["AB", "Alberta"],
    ["BC", "British Columbia"],
    ["MB", "Manitoba"],
    ["NB", "New Brunswick"],
    ["NL", "Newfoundland and Labrador"],
    ["NS", "Nova Scotia"],
    ["NT", "Northwest Territories"],
    ["NU", "Nunavut"],
    ["ON", "Ontario"],
    ["PE", "Prince Edward Island"],
    ["QC", "Quebec"],
    ["SK", "Saskatchewan"],
    ["YT", "Yukon Territory"]
];
this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

void getNewEmployee(string _error = null)
{
    render!("empnew.dt", deps, provs);
}

@errorDisplay!getNewEmployee
void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))

```

```

    {
        photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullpath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
    }
}

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.addEmployee(e);
redirect("list_employees");
}

void getListEmployees(string _error = null)
{
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps);
}

void getEditEmployee(BsonObjectID id, string _error = null)
{
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs);
}

@errorDisplay!getEditEmployee
void postEditEmployee(Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
}

```

```

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.editEmployee(e);
redirect("list_employees");
}

void getDeleteEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

void postDeleteEmployee(BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
void getShowEmployee(string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    render!("empshow.dt", e);
}

void getLogin()
{
    render!"login.dt";
}

void getEncryptPasswords()
{
    Admin[] admins = empModel.getAdmins;
    foreach(a; admins)
    {
        auto pword = createDigestPassword(realm, a.email, a.password);
        empModel.replacePassword(a.email, pword);
    }
    redirect("/");
}
}

```

And this is the state of source\empmongo.d so far.

```
module empmongo;
```

```
import vibe.db.mongo.mongo;

struct Employee
{
    BsonObjectID _id;
    string dept;
    string email;
    string pword;
    string fname;
    string lname;
    string phone;
    string paygd;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}

struct Admin
{
    string email;
    string password;
}

class EmployeeModel
{
    MongoCollection emptable, admintable;

    this()
    {
        auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
        emptable = empdb["employees"];
        admintable = empdb["admins"];
    }

    void addEmployee(Employee e)
    {
        emptable.insert
        ([
            "dept": e.dept,
            "email": e.email,
            "pword": e.pword,
            "fname": e.fname,
            "lname": e.lname,
            "phone": e.phone,
            "paygd": e.paygd,
            "photo": e.photo,
            "street": e.street,
            "city": e.city,
        ]);
    }
}
```

```

        "province": e.province,
        "postcode": e.postcode
    ]);
}

Employee[] getEmployees()
{
    Employee[] emps;
    auto results = emptable.find();
    foreach(doc;results) emps ~= deserializeBson!Employee(doc);
    return emps;
}

Employee getEmployee(BsonObjectID id)
{
    Employee emp;
    auto result = emptable.findOne(["_id":id]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}

void editEmployee(Employee e)
{
    emptable.update
    (
        ["_id":e._id],
        ["$set":
            [
                "dept": e.deprt,
                "email": e.email,
                "pword": e.pword,
                "fname": e.fname,
                "lname": e.lname,
                "phone": e.phone,
                "paygd": e.paygd,
                "photo": e.photo,
                "street": e.street,
                "city": e.city,
                "province": e.province,
                "postcode": e.postcode
            ]
        ]
    );
}

void deleteEmployee(BsonObjectID id)
{
    emptable.remove(["_id":id]);
}

```

```
Employee getEmployee(string first, string last)
{
    Employee emp;
    auto result = emptable.findOne(["fname":first, "lname":last]);
    if(!result.isNull) emp = deserializeBson!Employee(result);
    return emp;
}

Admin[] getAdmins()
{
    Admin[] admins;
    auto result = admintable.find();
    foreach(doc; result) admins ~= deserializeBson!Admin(doc);
    return admins;
}

void replacePassword(string email, string password)
{
    admintable.update(["email":email], ["$set":["password":password]]);
}
```

Now we let's turn to the authentication part of the log in system.

Authenticating the user

We are going to simplify things. We simply match the email and password combination received from the form to what we have on the admins collection to verify the user's identity. Since the saved password is encrypted, we also have to encrypt the raw password received from the form before comparing them.

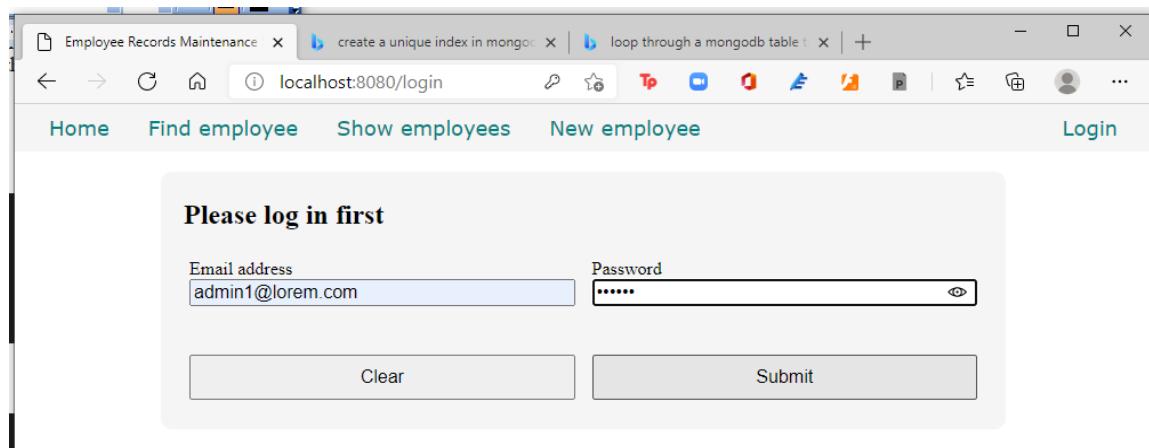
Edit source\empcontrol.d and add this method.

```
void postLogin(string email, string password)
{
    auto pword = createDigestPassword(realm, email, password);
    bool isAdmin = empModel.getAdmin(email, pword);
    if(isAdmin) redirect("list_employees");
    else redirect("/");
}
```

Edit source\empmongo.d and add this method.

```
bool getAdmin(string email, string password)
{
    auto result = admintable.findOne(["email":email, "password":password]);
    if(result.isNull) return false;
    return true;
}
```

Compile, run and refresh the browser, then try to log in.



Click the Submit button. If the list of employees is shown, you did good.

The screenshot shows a browser window with a tab titled "loop through a mongodb table". The main content area displays a table with the following data:

Department	First name	Last name	Phone number	Email address	Salary grade	Street address	City	Province	PostCode	Action	
Maintenance	Jen	Law	12345678909	jen@law.com	SG-1002	123 Front St	Vancouver	BC	V5A 2G4		
Shipping and Receiving	Angie	Jolie	12345678909	angie@jolie.com	SG-2012	89 J Perne St.	Toronto	ON	A1A 1A1		
Production	Anna	Bell	12345678909	anna@bell.com	SG-1234	123 Front St	Vancouver	BC	D5A 5F2		

Now let's save the login state to the session.

Saving the login state to the session

A session is when a user logs in up to the point when the user logs out. When we want to retrieve a particular record from the database, we simply use the `_id` field as the key to retrieve the record by passing the key to the next page.

When a user logs in, the log in state needs to be saved for the duration of the session so that the user can navigate from page to page without having to log in again and again.

If the user is logged in, we should save that state so that as the user navigates from page to page, the system can check if the user is authorized to access that particular page.

A variable, such as a logged-in state, can be saved to this session facility. Vibe.d has such a facility for saving session variables. It is called the session store, of which there are two kinds: `MemorySessionStore` and the database-based store. For now, we will use the `MemorySessionStore` kind.

Edit source\app.d.

```
import vibe.vibe;
import empcontrol;

void main()
{
    auto router = new URLRouter;
    router.get("*", serveStaticFiles("public/"));
    router.registerWebInterface(new EmployeeController);

    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];
    settings.sessionStore = new MemorySessionStore;

    listenHTTP(settings, router);
    runApplication();
}
```

Edit source\empcontrol.d.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;
```

```
struct User
{
    bool loggedIn;
    string email;
}

class EmployeeController
{
    private EmployeeModel empModel;
    private SessionVar!(User, "user") m_user;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ];
    string[][][] provs =
    [
        ["AB", "Alberta"],
        ["BC", "British Columbia"],
        ["MB", "Manitoba"],
        ["NB", "New Brunswick"],
        ["NL", "Newfoundland and Labrador"],
        ["NS", "Nova Scotia"],
        ["NT", "Northwest Territories"],
        ["NU", "Nunavut"],
        ["ON", "Ontario"],
        ["PE", "Prince Edward Island"],
        ["QC", "Quebec"],
        ["SK", "Saskatchewan"],
        ["YT", "Yukon Territory"]
    ];
    this()
    {
        empModel = new EmployeeModel;
    }

    void index()
    {
        render!("index.dt");
    }
}
```

```

void getNewEmployee(string _error = null)
{
    string error = _error;
    render!("empnew.dt", deps, provs, error);
}

@errorDisplay!getNewEmployee
void postNewEmployee(Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
    redirect("list_employees");
}

void getListEmployees(string _error = null)
{
    string error = _error;
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps, error);
}

void getEditEmployee(BsonObjectID id, string _error = null)
{
    string error = _error;
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
void postEditEmployee(Employee e)

```

```

{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}

void getDeleteEmployee(BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

void postDeleteEmployee(BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
void getShowEmployee(string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    enforce(e != Employee.init, "Employee not found!");
    render!("empshow.dt", e);
}

void getEncryptPasswords()
{
    Admin[] admins = empModel.getAdmins;
    foreach(a; admins)
}

```

```

    {
        auto pword = createDigestPassword(realm, a.email, a.password);
        empModel.replacePassword(a.email, pword);
    }
    redirect("/");
}

void getLogin(string _error = null)
{
    string error = _error;
    render!("login.dt", error);
}

@errorDisplay!getLogin
void postLogin(string email, string password)
{
    auto pword = createDigestPassword(realm, email, password);
    bool isAdmin = empModel.getAdmin(email, pword);
    enforce(isAdmin, "Email and password combination not found!");
    User user = m_user;
    user.loggedIn = true;
    user.email = email;
    m_user = user;
    redirect("list_employees");
}
}

```

We created a User struct to hold the logged-in state and will be a session variable.

```

struct User
{
    bool loggedIn;
    string email;
}

```

Then we declared a session variable named m_user to represent that struct.

```
private SessionVar!(User, "user") m_user;
```

We changed the getLogin() method to accept an _error variable from the postLogin() method if it encounters an error.

```
void getLogin(string _error = null)
```

We indicated that getLogin() will be called and passed the error if postLogin() encounters an error.

```
@errorDisplay!getLogin
```

And we also changed the value of `m_user` inside `postLogin()`, which starts the session.

```
User user = m_user;
user.loggedIn = true;
user.email = email;
m_user = user;
```

Edit `views\login.dt` to show any errors.

```
extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Please log in first
        form.form-grid(method="post", action="login")
            div.form-grid-column Email address<br />
                input.form-grid-column-
input(type="email", name="email", placeholder="email", required)
            div.form-grid-column Password<br />
                input.form-grid-column-
input(type="password", name="password", placeholder="password", required)
            div.form-grid-column
                br
                input.form-grid-column-button(type="reset", value="Clear")
            div.form-grid-column
                br
                input.form-grid-column-button(type="submit", value="Submit")
- if(error)
    div.error #{error}<br /><br />
```

Compile, run and refresh the browser. Click on the Login menu item and input a non-existing admin user.

The screenshot shows a web application interface. At the top, there's a navigation bar with the word "Employee". Below it, a header contains links for "Show employees" and "New employee". A modal dialog box is centered on the screen with a title "Please log in first". Inside the dialog, there are two input fields: one for "Email address" containing "su@ma.com" and another for "Password" containing "*****". Below the inputs are two buttons: "Clear" on the left and "Submit" on the right. The overall layout is clean and modern.

And you get this error.

[Employee](#) [Show employees](#) [New employee](#)

Please log in first

Email address

email

Password

password

Email and password combination not found!

Which means the error-trapping system is working.

Next is how to let the other pages know about the logged-in state.

Enforcing authorization with the session variable

We want to restrict access to all the pages in the website except the home page and the login page, but right now, clicking on any of the links on the menu shows the page. Meaning, all the links in the menu are non-restrictive and is viewable by anyone. We need to rectify that.

Only the home page and the login page should be accessible to anyone and the rest should be accessible only to logged-in users. There should be a way to check if the user is logged in or not before deciding to open the page.

Edit source\empcontrol.d and add at the end.

```
private enum auth = before!ensureAuth("_authUser");

private string ensureAuth(HTTPSRequest req, HTTPSResponse res)
{
    if(!m_user.loggedIn) redirect("login");
    return m_user.email;
}
mixin PrivateAccessProxy;
```

The @auth annotation is a shortcut for calling the ensureAuth() function to check if the user is logged in before running a method.

That mixin statement there is needed to make this private function accessible.

This function redirects to the getLogin() method if the user is not logged in yet.

Then we defined a shortcut to the ensureAuth() function with this:

```
private enum auth = before!ensureAuth("_authUser");
```

so we can just use @auth to mean we are calling the ensureAuth() private function, like this:

```
@auth
void getNewEmployee(string _error = null)
```

The @auth annotation means call the ensureAuth() function, which checks the logged-in state, before running this getNewEmployee() method.

Since the ensureAuth() function is passing the _authUser variable, we now have to add it as an argument to all the methods that call ensureAuth(), like this:

```
@auth
```

```
void getNewEmployee(string _authUser, string _error = null)
```

So here is the full source\empcontrol.d file.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmysql;

struct User
{
    bool loggedIn;
    string email;
}

class EmployeeController
{
    private EmployeeModel empModel;
    private SessionVar!(User, "user") m_user;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ];
    string[][] provs =
    [
        ["AB", "Alberta"],
        ["BC", "British Columbia"],
        ["MB", "Manitoba"],
        ["NB", "New Brunswick"],
        ["NL", "Newfoundland and Labrador"],
        ["NS", "Nova Scotia"],
        ["NT", "Northwest Territories"],
        ["NU", "Nunavut"],
        ["ON", "Ontario"],
        ["PE", "Prince Edward Island"],
        ["QC", "Quebec"]
    ];
}
```

```

        [ "SK", "Saskatchewan"],
        [ "YT", "Yukon Territory"]
    ];

this()
{
    empModel = new EmployeeModel;
}

void index()
{
    render!("index.dt");
}

@auth
void getNewEmployee(string _authUser, string _error = null)
{
    string error = _error;
    render!("empnew.dt", deps, provs, error);
}

@errorDisplay!getNewEmployee
@auth
void postNewEmployee(string _authUser, Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
    redirect("list_employees");
}

```

```

@auth
void getListEmployees(string _authUser, string _error = null)
{
    string error = _error;
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps, error);
}

@auth
void getEditEmployee(string _authUser, int id, string _error = null)
{
    string error = _error;
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
@auth
void postEditEmployee(string _authUser, Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}

@auth
void getDeleteEmployee(string _authUser, int id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

```

```

}

@auth
void postDeleteEmployee(string _authUser, int id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
@auth
void getShowEmployee(string _authUser, string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    enforce(e != Employee.init, "Employee not found!");
    render!("empshow.dt", e);
}

void getLogin(string _error = null)
{
    string error = _error;
    render!("login.dt", error);
}

@errorDisplay!getLogin
void postLogin(string email, string password)
{
    auto pword = createDigestPassword(realm, email, password);
    bool isAdmin = empModel.getAdmin(email, pword);
    enforce(isAdmin, "Email and password combination not found!");
    User user = m_user;
    user.loggedIn = true;
    user.email = email;
    m_user = user;
    redirect("list_employees");
}

private enum auth = before!ensureAuth("_authUser");

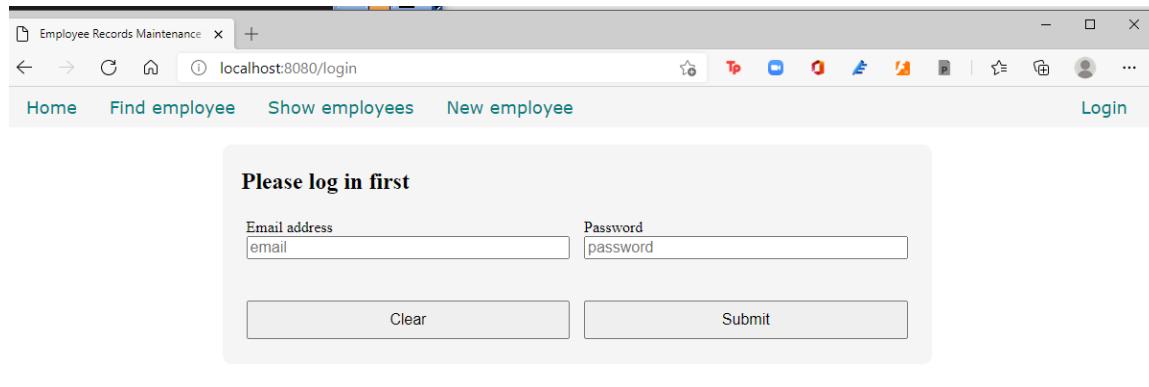
private string ensureAuth(HTTPRequest req, HTTPResponse res)
{
    if(!m_user.loggedIn) redirect("login");
    return m_user.email;
}
mixin PrivateAccessProxy;
}

```

We added the @auth annotation for each method that requires authorization.

We did not make any changes to the source\empmongo.d, so we are good.

Compile, run and refresh the browser. Click any link on the menu except the Home link and you will be redirected to the login page.



Now we are assured that sensitive data is protected and accessible only to authorized users.

How about logging out?

Let's just decide that if the user clicks on the Home link on the menu, the user is logged out.

Let's edit the index() method then.

```
void index()
{
    m_user = User.init;
    terminateSession;
    render!("index.dt");
}
```

Here, we re-initialized the m_user session variable then explicitly terminated the session before displaying the index.dt template page.

Compile, run, refresh the browser and test the app. You will see that when you click on the Home link, you have to log in again to see the other pages.

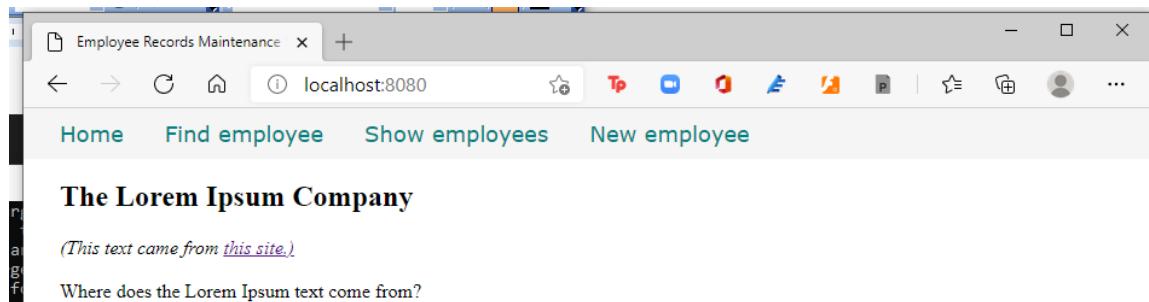
Now, since the login page automatically shows up if the user is not logged in yet, the Login link on the menu looks superfluous. We can get rid of it then.

Edit views\menu.dt and remove the Login link.

```
div.menu-item
```

```
a.item-link(href="/") Home
div.menu-item
  a.item-link(href="#find") Find employee
div.menu-item
  a.item-link(href="list_employees") Show employees
div.menu-item
  a.item-link(href="new_employee") New employee
div#find.modal-form
  div.modal-form-wrapper-find
    form.modal-form-grid(method="get", action="show_employee")
      input.text-
control(name="fname", type="text", placeholder=" First name", required)
  input.text-
control(name="lname", type="text", placeholder=" Last name", required)
  input#but-reset(type="reset", value="Clear")
  input#but-submit(type="submit", value="Submit")
a.close(href="#close") Cancel
```

Compile, run, refresh the browser and test the app again. This time, no more Login link.



With that, we are done with our project using MongoDB. Hurray!

All the source code

So here is the full source\app.d.

```
import vibe.vibe;
import empcontrol;

void main()
{
    auto router = new URLRouter;
    router.get("*", serveStaticFiles("public/"));
    router.registerWebInterface(new EmployeeController);

    auto settings = new HTTPServerSettings;
    settings.port = 8080;
    settings.bindAddresses = [":1", "127.0.0.1"];
    settings.sessionStore = new MemorySessionStore;

    listenHTTP(settings, router);
    runApplication();
}
```

And here is the full source\empcontrol.d.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmongo;

struct User
{
    bool loggedIn;
    string email;
}

class EmployeeController
{
    private EmployeeModel empModel;
    private SessionVar!(User, "user") m_user;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
    ]
}
```

```

    "Maintenance",
    "Shipping and Receiving",
    "Purchasing and Supplies",
    "IT Services",
    "Human Resources",
    "Marketing"
];
string[][] provs =
[
    ["AB", "Alberta"],
    ["BC", "British Columbia"],
    ["MB", "Manitoba"],
    ["NB", "New Brunswick"],
    ["NL", "Newfoundland and Labrador"],
    ["NS", "Nova Scotia"],
    ["NT", "Northwest Territories"],
    ["NU", "Nunavut"],
    ["ON", "Ontario"],
    ["PE", "Prince Edward Island"],
    ["QC", "Quebec"],
    ["SK", "Saskatchewan"],
    ["YT", "Yukon Territory"]
];
this()
{
    empModel = new EmployeeModel;
}

void index()
{
    m_user = User.init;
    terminateSession;
    render!("index.dt");
}

@auth
void getNewEmployee(string _authUser, string _error = null)
{
    string error = _error;
    render!("empnew.dt", deps, provs, error);
}

@errorDisplay!getNewEmployee
@auth
void postNewEmployee(string _authUser, Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)

```

```

{
    string ext = extension(pic.filename.name);
    string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
    if(canFind(exts, ext))
    {
        photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullpath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
    }
}
e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.addEmployee(e);
redirect("list_employees");
}

@auth
void getListEmployees(string _authUser, string _error = null)
{
    string error = _error;
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps, error);
}

@auth
void getEditEmployee(string _authUser, BsonObjectID id, string _error = null)
{
    string error = _error;
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
@auth
void postEditEmployee(string _authUser, Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
}

```

```

        photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
        string dir = "./public/uploads/photos/";
        mkdirRecurse(dir);
        string fullPath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
        try moveFile(pic.tempPath, NativePath(fullPath));
        catch (Exception ex) copyFile(pic.tempPath, NativePath(fullPath));
    }
}

e.photo = photopath;
if(e.phone.length == 0) e.phone = "(123) 456 7890";
if(e.paygd.length == 0) e.paygd = "none yet";
if(e.postcode.length == 0) e.postcode = "A1A 1A1";
e.pword = createDigestPassword(realm, e.email, e.pword);
empModel.editEmployee(e);
redirect("list_employees");
}

@auth
void getDeleteEmployee(string _authUser, BsonObjectID id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

@auth
void postDeleteEmployee(string _authUser, BsonObjectID id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
@auth
void getShowEmployee(string _authUser, string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    enforce(e != Employee.init, "Employee not found!");
    render!("empshow.dt", e);
}

void getLogin(string _error = null)
{
    string error = _error;
    render!("login.dt", error);
}

@errorDisplay!getLogin
void postLogin(string email, string password)
{
    auto pword = createDigestPassword(realm, email, password);
}

```

```

        bool isAdmin = empModel.getAdmin(email, pword);
        enforce(isAdmin, "Email and password combination not found!");
        User user = m_user;
        user.loggedIn = true;
        user.email = email;
        m_user = user;
        redirect("list_employees");
    }

    private enum auth = before!ensureAuth("_authUser");

    private string ensureAuth(HTTPRequest req, HTTPServerResponse res)
    {
        if(!m_user.loggedIn) redirect("login");
        return m_user.email;
    }
    mixin PrivateAccessProxy;
}

```

Here is the full source\empmongo.d.

```

module empmongo;

import vibe.db.mongo.mongo;

struct Employee
{
    BsonObjectID _id;
    string dept;
    string email;
    string pword;
    string fname;
    string lname;
    string phone;
    string paygd;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}

struct Admin
{
    string email;
    string password;
}

class EmployeeModel

```

```

{
    MongoCollection emptable, admintable;

    this()
    {
        auto empdb = connectMongoDB("127.0.0.1").getDatabase("empdb");
        emptable = empdb["employees"];
        admintable = empdb["admins"];
    }

    void addEmployee(Employee e)
    {
        emptable.insert
        (
            [
                "deprt": e.deprt,
                "email": e.email,
                "pword": e.pword,
                "fname": e.fname,
                "lname": e.lname,
                "phone": e.phone,
                "paygd": e.paygd,
                "photo": e.photo,
                "street": e.street,
                "city": e.city,
                "province": e.province,
                "postcode": e.postcode
            ]);
    }

    Employee[] getEmployees()
    {
        Employee[] emps;
        auto results = emptable.find();
        foreach(doc;results) emps ~= deserializeBson!Employee(doc);
        return emps;
    }

    Employee getEmployee(BsonObjectID id)
    {
        Employee emp;
        auto result = emptable.findOne(["_id":id]);
        if(!result.isNull) emp = deserializeBson!Employee(result);
        return emp;
    }

    void editEmployee(Employee e)
    {
        emptable.update
        (
            ["_id":e._id],

```

```

[ "$set": [
  [
    "dept": e.deprt,
    "email": e.email,
    "pword": e.pword,
    "fname": e.fname,
    "lname": e.lname,
    "phone": e.phone,
    "paygd": e.paygd,
    "photo": e.photo,
    "street": e.street,
    "city": e.city,
    "province": e.province,
    "postcode": e.postcode
  ]
]
);
}

void deleteEmployee(BsonObjectID id)
{
  emptable.remove(["_id":id]);
}

Employee getEmployee(string first, string last)
{
  Employee emp;
  auto result = emptable.findOne(["fname":first, "lname":last]);
  if(!result.isNull) emp = deserializeBson!Employee(result);
  return emp;
}

Admin[] getAdmins()
{
  Admin[] admins;
  auto result = admintable.find();
  foreach(doc; result) admins ~= deserializeBson!Admin(doc);
  return admins;
}

void replacePassword(string email, string password)
{
  admintable.update(["email":email], ["$set":["password":password]]);
}

bool getAdmin(string email, string password)
{
  auto result = admintable.findOne(["email":email, "password":password]);
  if(result.isNull) return false;
  return true;
}

```

```
    }  
}
```

The cssformgrid.dt.

```
:css  
.form-grid-wrapper  
{  
    width: 700px;  
    height: auto;  
    margin: 20px auto;  
    padding: 5px 20px;  
    border-radius: 10px;  
    background-color: whitesmoke;  
}  
.form-grid  
{  
    display: grid;  
    grid-template-columns: 1fr 1fr;  
    gap: 5px;  
}  
.form-grid-column  
{  
    display: inline;  
    height: auto;  
    padding: 5px;  
}  
.form-grid-column-field  
{  
    width: 100%;  
    font-size: 16px;  
    border-bottom: 1px solid black;  
    padding: 5px 0;  
}  
.form-grid-column-button  
{  
    width: 100%;  
    font-size: 16px;  
    height: 40px;  
    margin-top: 10px;  
}  
.form-grid-column-input  
{  
    width: 100%;  
    font-size: 16px;  
}
```

The cssjs.dt.

```
:css
body
{
    margin: 0;
    padding: 0;
}
.container
{
    display: grid;
    grid-template-rows: 40px auto 30px;
    height: 100%;
    width: 100%;
    margin: 0;
    padding: 0;
}
.error
{
    color: brown;
    font-weight: bold;
}
.text-right
{
    text-align: right;
}
/*menu styles***** */
.menu
{
    position: fixed;
    top: 0;
    width: 100%;
    padding: 10px 0;
    background-color: whitesmoke;
}
.menu-item
{
    display: inline;
}
.item-link
{
    margin-left: 30px;
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    font-size: 18px;
    color: teal;
    text-decoration: none;
}
.item-link-right
{
    float: right;
    margin-right: 30px;
}
```

```
.modal-form
{
    position: fixed;
    font-family:'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    top: 0;
    right: 0;
    bottom: 0;
    left: 0;
    background: rgba(0,0,0,0.5);
    z-index: 99999;
    opacity: 0;
    pointer-events: none;
}
#find:target
{
    opacity: 1;
    pointer-events: auto;
}
.modal-form-grid {
    display: grid;
    grid-template: 30px 30px 30px / 1fr 1fr;
    grid-gap: 10px;
}
.text-control
{
    grid-column: 1 / 3;
}
#but-reset
{
    grid-column: 1 / 2;
}
#but-submit
{
    grid-column: 2 / 3;
}
.modal-form-wrapper-login
{
    width: 250px;
    position: absolute;
    right: 0;
    margin-top: 40px;
    padding: 25px 20px;
    border-radius: 10px;
    text-align: center;
    background-color: whitesmoke;
}
.modal-form-wrapper-find
{
    width: 250px;
    position: relative;
```

```

        left: 120px;
        margin-top: 40px;
        padding: 25px 20px;
        border-radius: 10px;
        text-align: center;
        background-color: whitesmoke;
    }
/*end of menu styles*/
/*content styles******/
.content
{
    padding: 40px 30px;
}
/*end of content styles*/
/*footer styles******/
.footer
{
    position: fixed;
    bottom: 0;
    width: 100%;
    background-color: whitesmoke;
    padding: 5px 0;
}
.copyright
{
    font-family: Verdana, Geneva, Tahoma, sans-serif;
    font-size: 14px;
    text-align: center;
    color: teal;
}
.form-hidden
{
    padding: 0;
    margin: 0;
    display: inline;
}

```

The csstable.dt.

```

:css
.table-wrapper
{
    margin: 20px auto;
    padding: 20px;
}
table
{
    margin: 0 auto;
    padding: 20px;

```

```

    border-collapse: collapse;
}
table td, table th
{
    border: 1px solid black;
    margin: 0;
    padding: 0 5px;
}
.no-border
{
    border: 0;
}

```

The empdelete.dt.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2.brown Are you sure you want to delete this record?
        form.form-grid(method="post", action="delete_employee")
            div.form-grid-column
                | Department:<br />
                div.form-grid-column-field #{e.deprt}
            div.form-grid-column
                | Salary grade:<br />
                div.form-grid-column-field #{e.paygd}
            div.form-grid-column
                | Email address:<br />
                div.form-grid-column-field #{e.email}
            div.form-grid-column
                | Password:<br />
                div.form-grid-column-field *****
            div.form-grid-column
                | First name:<br />
                div.form-grid-column-field #{e.fname}
            div.form-grid-column
                | Last name:<br />
                div.form-grid-column-field #{e.lname}
            div.form-grid-column
                | Phone:<br />
                div.form-grid-column-field #{e.phone}
            div.form-grid-column
                | Street address (without city):<br />
                div.form-grid-column-field #{e.street}
            div.form-grid-column
                | City:<br />
                div.form-grid-column-field #{e.city}
            div.form-grid-column

```

```

| Province:<br />
div.form-grid-column-field #{e.province}
div.form-grid-column
| Postal code:<br />
div.form-grid-column-field #{e.postcode}
div.form-grid-column
| Profile picture:<br />
img(src="#{e.photo}", height="100px")
input(type="hidden", name="id", value="#{e._id}")
div.form-grid-column
a(href="list_employees")
button.form-grid-column-button(type="button") No
div.form-grid-column
input.form-grid-column-button(type="submit", value="Yes")

```

The empedit.dt.

```

extends layout
block maincontent
  include cssformgrid.dt
  div.form-grid-wrapper
    h2 Edit employee details
    form.form-grid(method="post", action="edit_employee", enctype="multipart/form-data")
      div.form-grid-column Department<br />
        select#depid.form-grid-column-input(name="e_deprt", value="#{e.deprt}")
          - foreach(dep; deps)
            - if(dep == e.deprt)
              option(value="#{dep}", selected) #{dep}
            - else
              option(value="#{dep}") #{dep}
      div.form-grid-column Salary grade<br />
        input.form-grid-column-
      input(type="text", name="e_paygd", value="#{e.paygd}")
      div.form-grid-column Email address<br />
        input.form-grid-column-
      input(type="email", name="e_email", value="#{e.email}")
      div.form-grid-column Password<br />
        input.form-grid-column-
      input(type="password", name="e_pword", value="#{e.pword}")
      div.form-grid-column First name<br />
        input.form-grid-column-
      input(type="text", name="e_fname", value="#{e.fname}")
      div.form-grid-column Last name<br />
        input.form-grid-column-
      input(type="text", name="e_lname", value="#{e.lname}")
      div.form-grid-column Phone<br />
        input.form-grid-column-
      input(type="text", name="e_phone", value="#{e.phone}")

```

```



```

The emplist.dt.

```

extends layout
block maincontent
    include csstable.dt
    div.table-wrapper
        table
            tr
                td.no-border(colspan="11")
                    - if(error)
                        span.error #{error}<br /><br />
            tr
                th Department
                th First name
                th Last name
                th Phone number
                th Email address

```

```

th Salary grade
th Street address
th City
th Province
th PostCode
th Action
- foreach(e; emps)
tr
    td #{e.deprt}
    td #{e.fname}
    td #{e.lname}
    td #{e.phone}
    td #{e.email}
    td #{e.paygd}
    td #{e.street}
    td #{e.city}
    td #{e.province}
    td #{e.postcode}
    td &nbsp;
        form.form-hidden(method="get", action="edit_employee")
            input(type="hidden", name="id", value="#{e._id}")
            input(type="image", src="images/pen.ico")
        | &nbsp;
        form.form-hidden(method="get", action="delete_employee")
            input(type="hidden", name="id", value="#{e._id}")
            input(type="image", src="images/trash.ico")
        | &nbsp;

```

The empnew.dt.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 New employee details
        form.form-grid(method="post", action="new_employee", enctype="multipart/form-data")
            div.form-grid-column Department<br />
                select#dept.form-grid-column-input(name="e_deprt")
                    - foreach(dep; deps)
                        option(value="#{dep}") #{dep}
            div.form-grid-column Salary grade<br />
                input.form-grid-column-
input(type="text", name="e_paygd", placeholder="Salary grade")
            div.form-grid-column Email address<br />
                input.form-grid-column-
input(type="email", name="e_email", placeholder="email@address.com", required)
            div.form-grid-column Password<br />

```

```

        input.form-grid-column-
input(type="password", name="e_pword", placeholder="password", required)
    div.form-grid-column First name<br />
        input.form-grid-column-
input(type="text", name="e_fname", placeholder="First name", required)
    div.form-grid-column Last name<br />
        input.form-grid-column-
input(type="text", name="e_lname", placeholder="Last name", required)
    div.form-grid-column Phone<br />
        input.form-grid-column-
input(type="text", name="e_phone", placeholder="Phone number")
    div.form-grid-column Street address (without city)<br />
        input.form-grid-column-
input(type="text", name="e_street", placeholder="Street address", required)
    div.form-grid-column City<br />
        input.form-grid-column-
input(type="text", name="e_city", placeholder="City", required)
    div.form-grid-column Province<br />
        select#province.form-grid-column-input(name="e_province")
            - foreach(prov; provs)
                option(value="#{prov[0]}") #{prov[1]}
    div.form-grid-column Postal code<br />
        input.form-grid-column-
input(type="text", name="e_postcode", placeholder="A1A 1A1")
    div.form-grid-column ID Picture<br />
        input.form-grid-column-input(type="file", name="picture")
    input(type="hidden", name="e_photo")
    input(type="hidden", name="e_id", value="123456789012345678901234")
    div.form-grid-column
        br
        input.form-grid-column-button(type="reset", value="Clear")
    div.form-grid-column
        br
        input.form-grid-column-button(type="submit", value="Submit")
- if(error)
    div.error #{error}

```

The empshow.dt.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Employee details
        form.form-grid(method="get", action="edit_employee")
            div.form-grid-column
                | Department<br />
                div.form-grid-column-field #{e.deprt}
            div.form-grid-column

```

```

| Salary grade<br />


```

The index.dt.

```

extends layout
block maincontent
h2 The Lorem Ipsum Company
div
i (This text came from <a href="https://www.lipsum.com/" target="_blank" rel="noopener noreferrer">this site.)</a><br /><br />

```

```


Where does the Lorem Ipsum text come from?<br /><br />
    Contrary to popular belief, Lorem Ipsum is not simply random text.
    It has roots in a piece of classical Latin literature from 45 BC,
    making it over 2000 years old. Richard McClintock, a Latin
    professor at Hampden-Sydney College in Virginia, looked up one of
    the more obscure Latin words, consectetur, from a Lorem Ipsum
    passage, and going through the cites of the word in classical
    literature, discovered the undoubtable source. Lorem Ipsum comes
    from sections 1.10.32 and 1.10.33 of "de Finibus Bonorum et
    Malorum" (The Extremes of Good and Evil) by Cicero, written in 45
    BC. This book is a treatise on the theory of ethics, very popular
    during the Renaissance. The first line of Lorem Ipsum, "Lorem
    ipsum dolor sit amet..", comes from a line in section 1.10.32.
<br /><br /><br />
    Why do we use it?<br /><br />
    It is a long established fact that a reader will be distracted by
    the readable content of a page when looking at its layout. The
    point of using Lorem Ipsum is that it has a more-or-less normal
    distribution of letters, as opposed to using 'Content here,
    content here', making it look like readable English. Many desktop
    publishing packages and web page editors now use Lorem Ipsum as
    their default model text, and a search for 'lorem ipsum' will
    uncover many web sites still in their infancy. Various versions
    have evolved over the years, sometimes by accident, sometimes on
    purpose (injected humour and the like).
<br /><br /><br />


```

The login.dt.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Please log in first
        form.form-grid(method="post", action="login")
            div.form-grid-column Email address<br />
                input.form-grid-column-
input(type="email", name="email", placeholder="email", required)
            div.form-grid-column Password<br />
                input.form-grid-column-
input(type="password", name="password", placeholder="password", required)
            div.form-grid-column
                br
                input.form-grid-column-button(type="reset", value="Clear")
            div.form-grid-column
                br
                input.form-grid-column-button(type="submit", value="Submit")
- if(error)
```

```
div.error #{error}<br /><br />
```

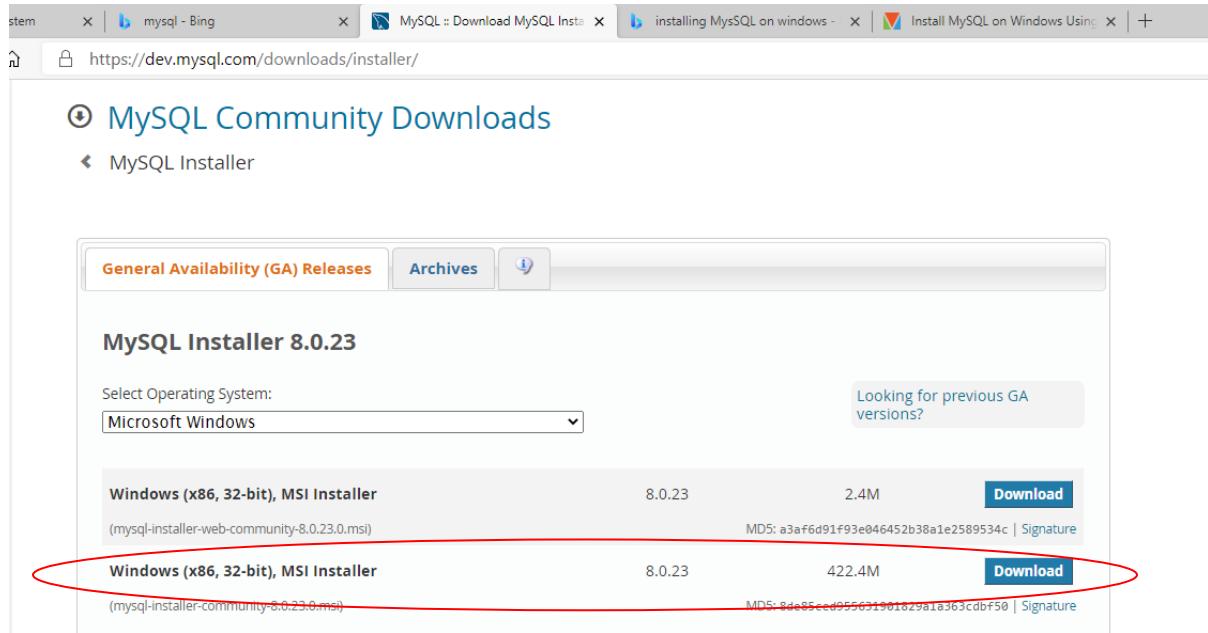
Time to port the whole thing to MySQL.

Setting up the MySQL server and tools

As of this writing (February 27, 2021), the Vibe.d driver for Mysql, mysql-native, is not working for Windows since their last upgrade, so I did all of these in my Ubuntu machine. Here's hoping that the MySQL driver is fixed before you read this.

Download the MySQL community server here:

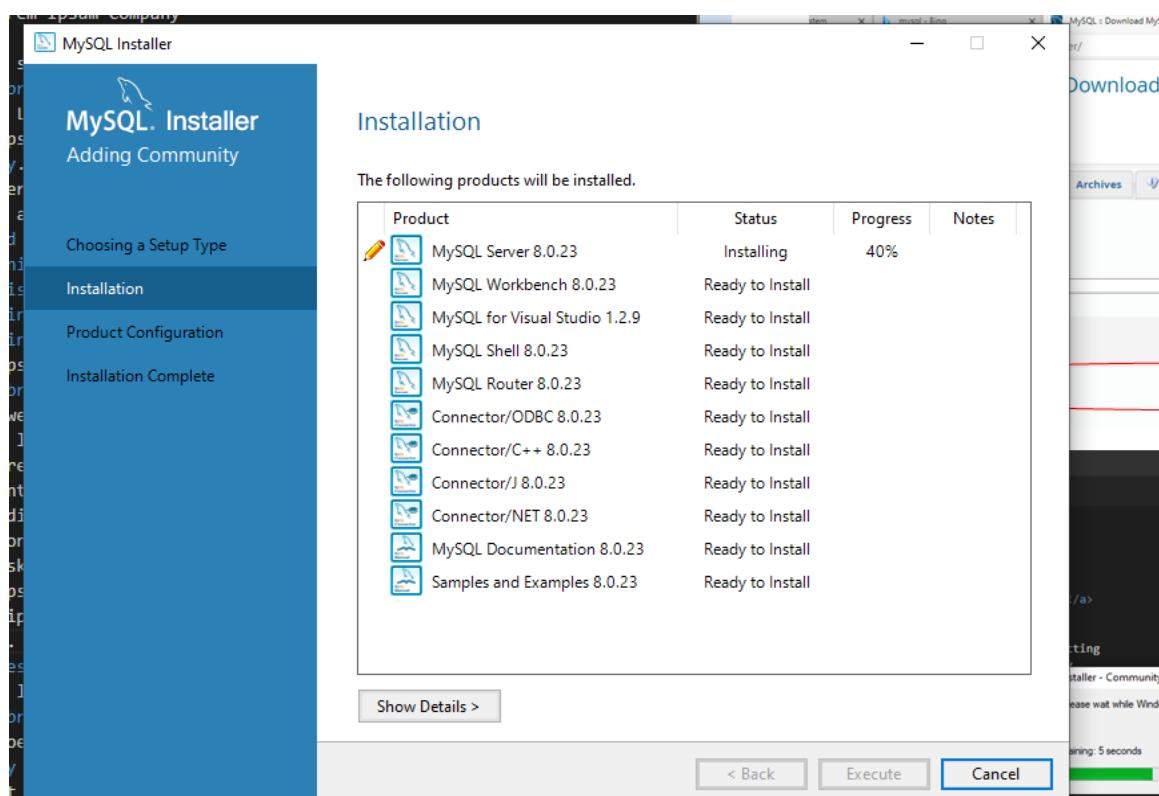
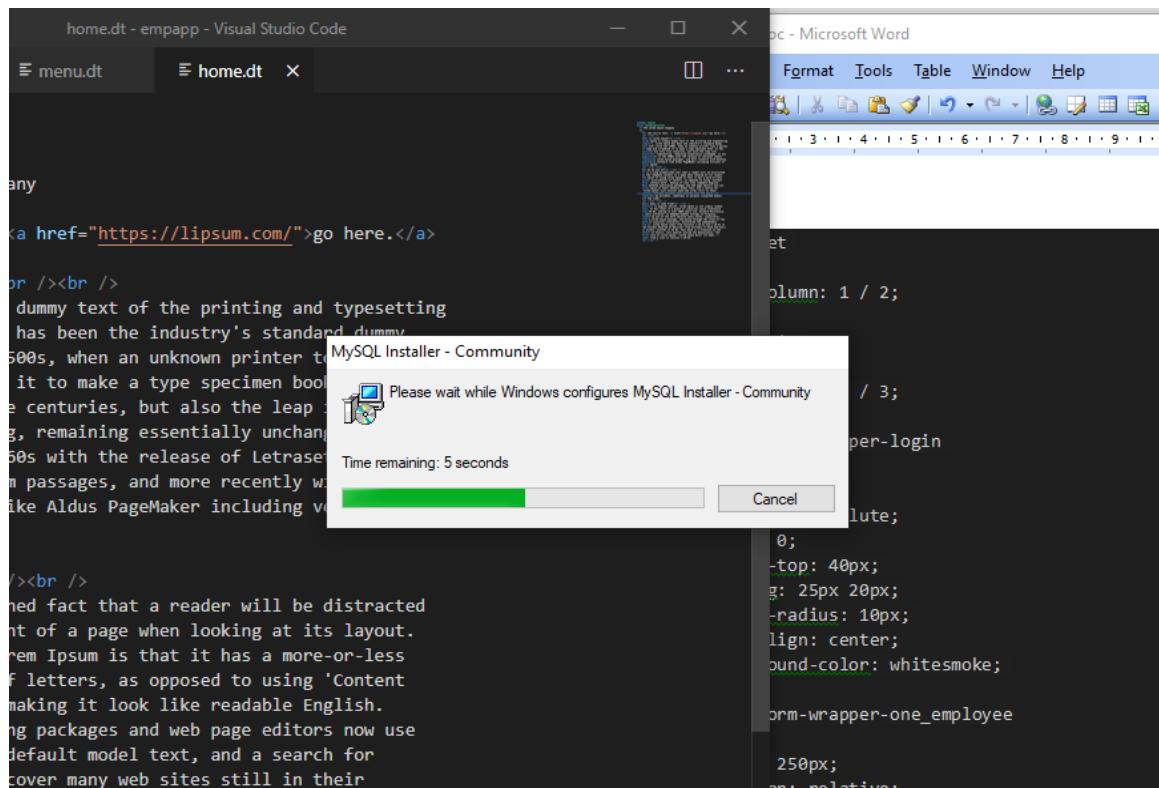
[MySQL :: Download MySQL Installer](https://dev.mysql.com/downloads/installer/)

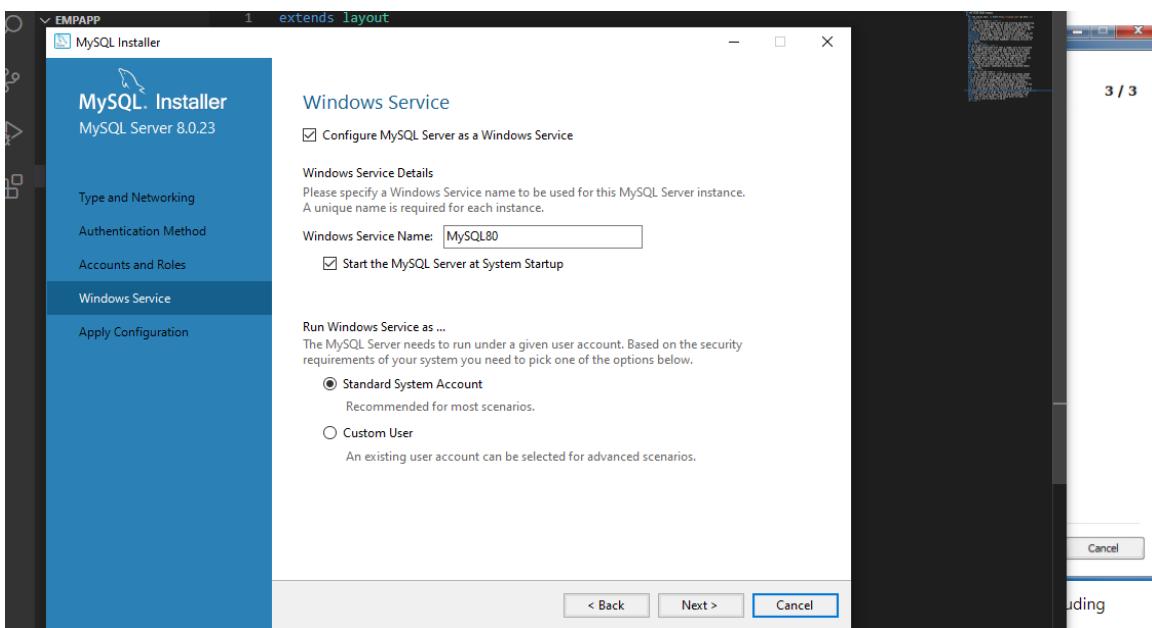
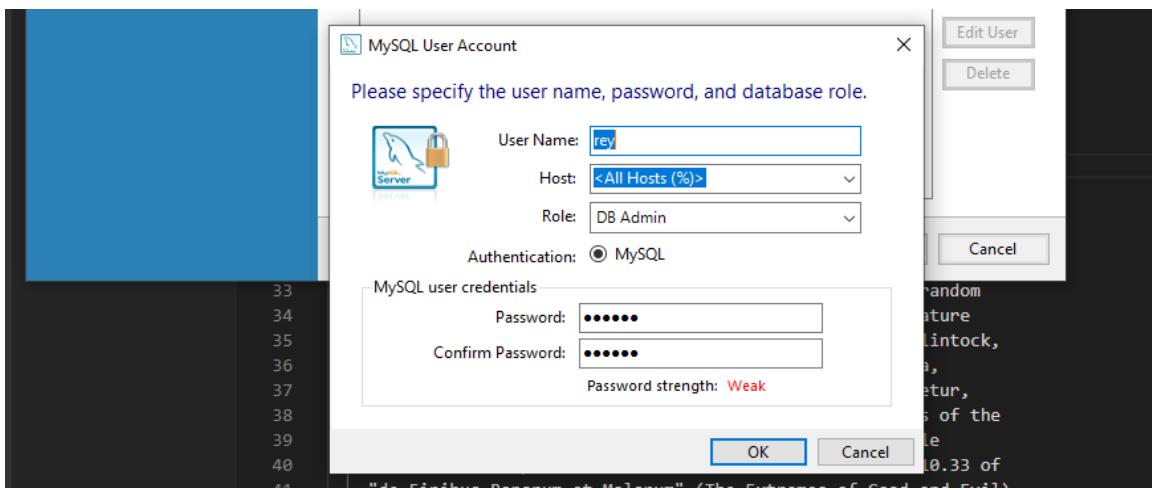


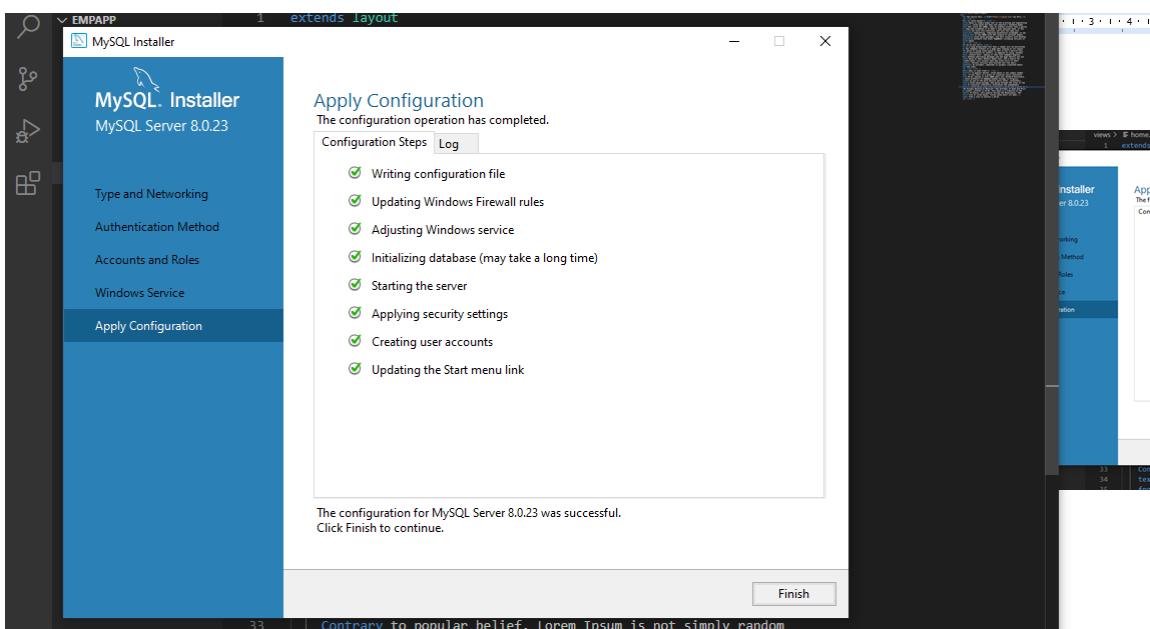
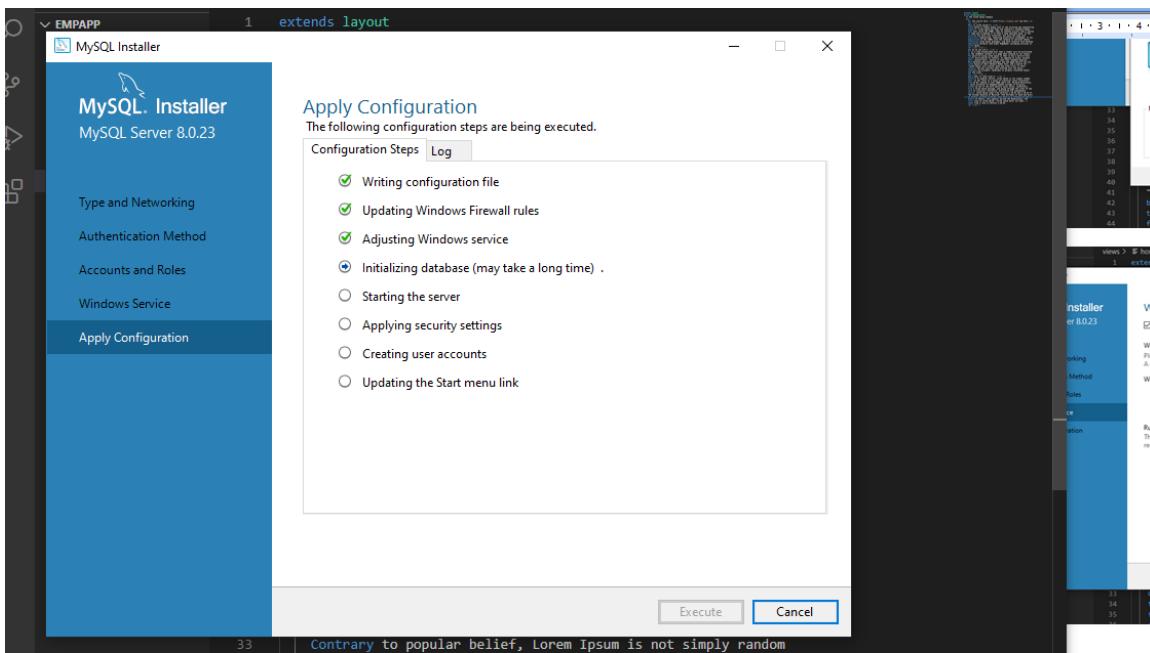
The screenshot shows a web browser window with multiple tabs open. The active tab is titled "MySQL :: Download MySQL Installer". The URL in the address bar is <https://dev.mysql.com/downloads/installer/>. The page content is the "MySQL Community Downloads" section, specifically for the "MySQL Installer 8.0.23". There are two download links listed:

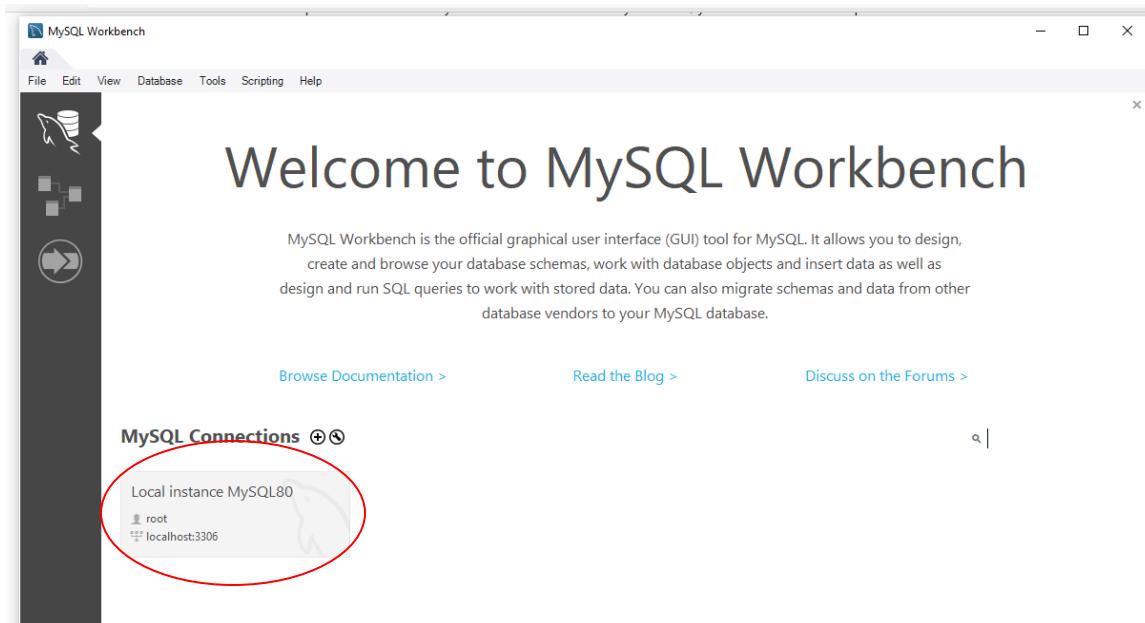
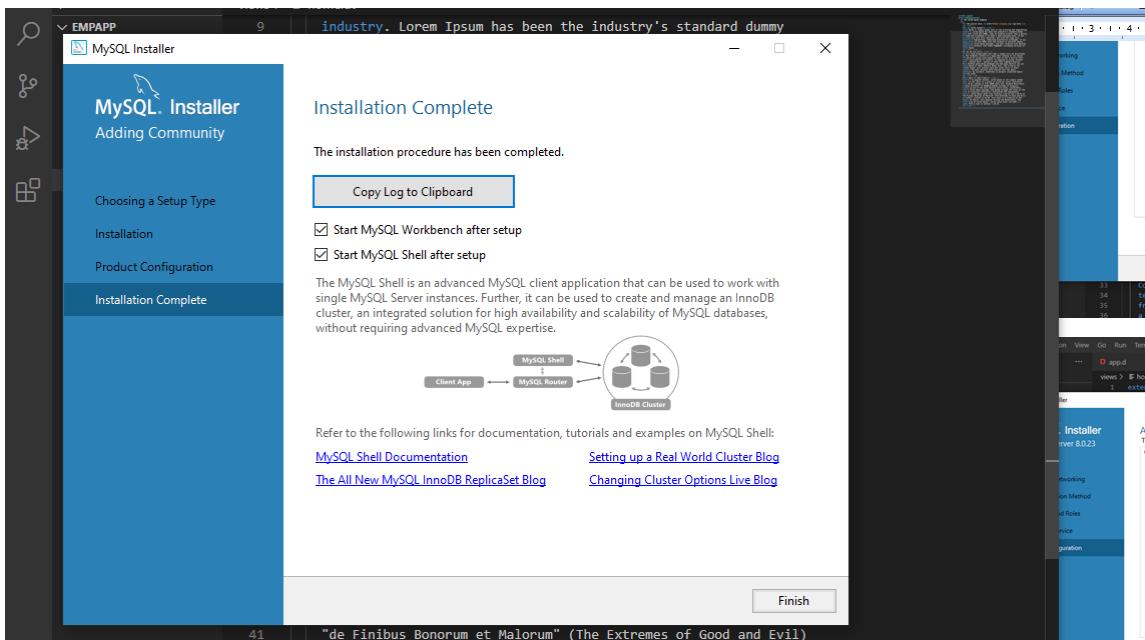
- Windows (x86, 32-bit), MSI Installer** (mysql-installer-web-community-8.0.23.0.msi)
File size: 2.4M
[Download](#) (button circled in red)
- Windows (x86, 32-bit), MSI Installer** (mysql-installer-community-8.0.23.0.msi)
File size: 422.4M
[Download](#) (button circled in red)

Other tabs visible in the browser include "stem", "mysql - Bing", "installing MySQL on windows -", and "Install MySQL on Windows Using".

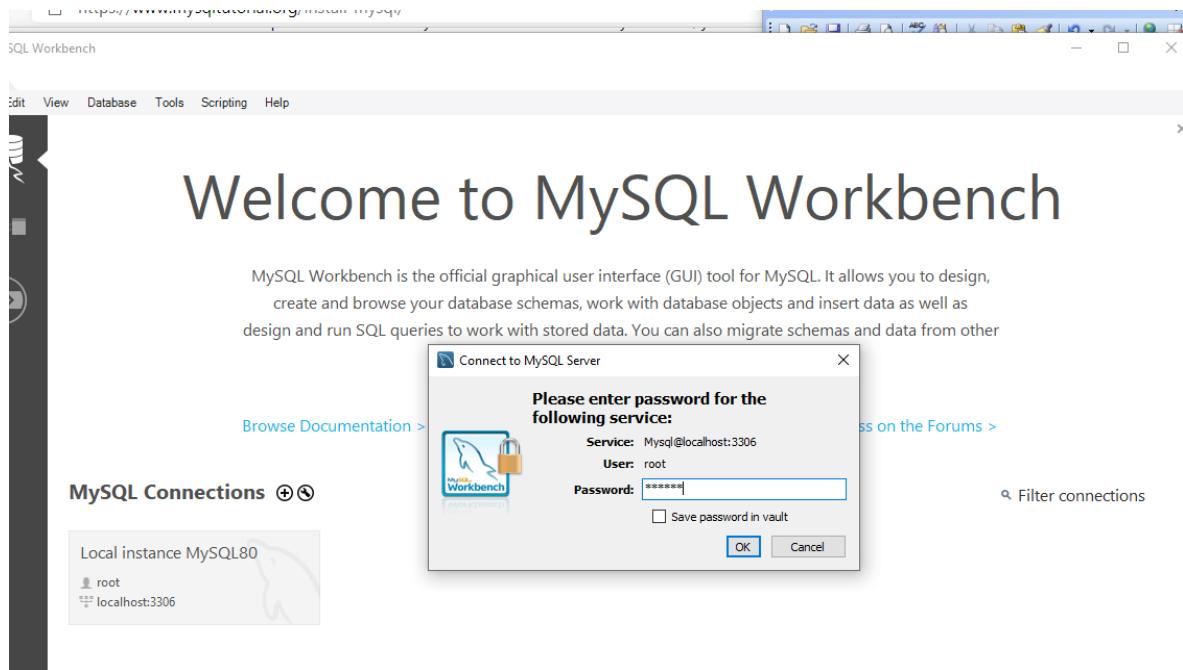








Click on the Local instance MySQL80 shaded area to open the password dialog.

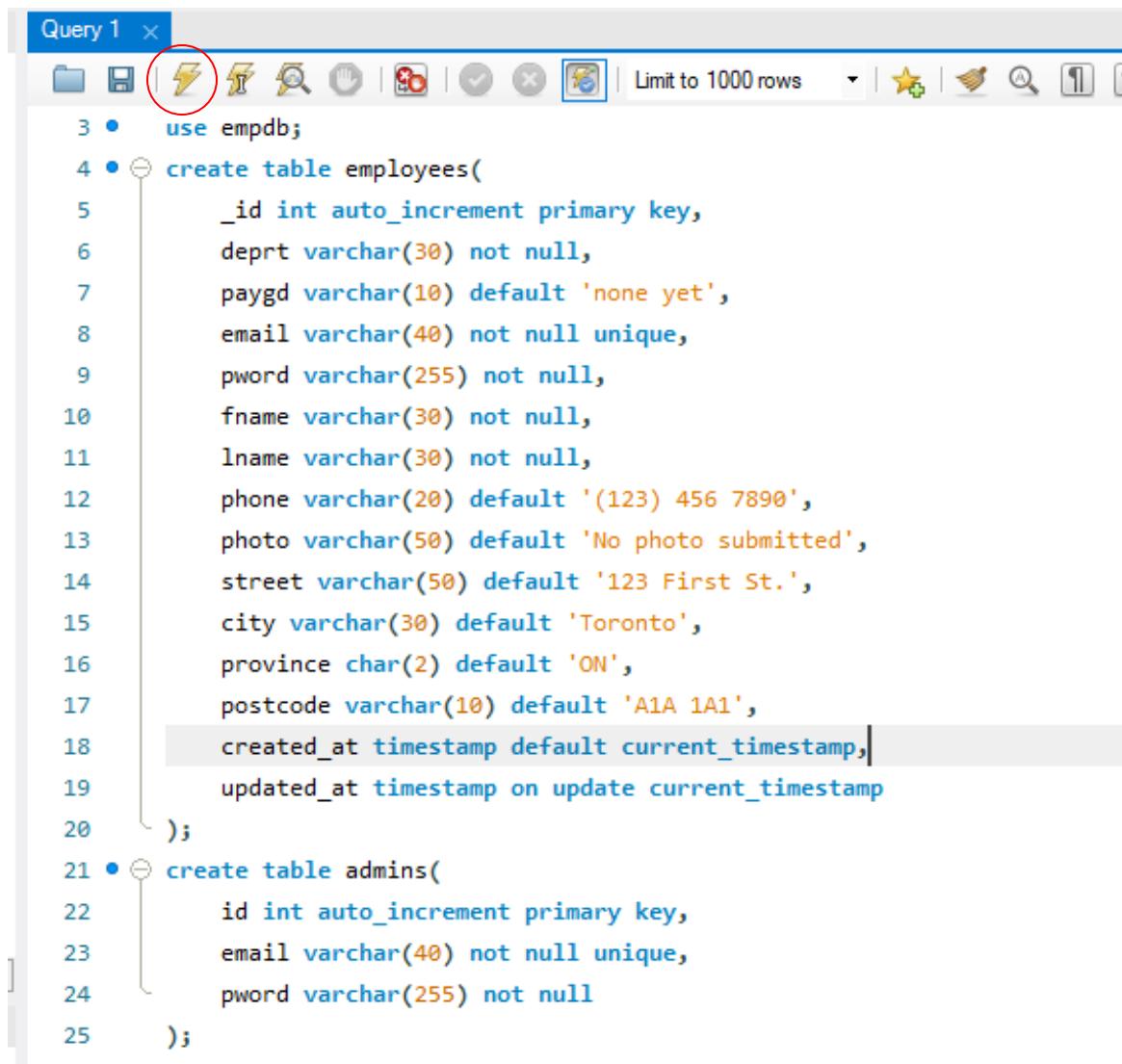


Type the root password you set earlier and click OK. The MySQL Workbench front page opens.

Let's use this schema.

```
create database empdb;
use empdb;
create table employees(
    _id int auto_increment primary key,
    dept varchar(30) not null,
    paygd varchar(10) default 'none yet',
    email varchar(40) not null unique,
    pword varchar(255) not null,
    fname varchar(30) not null,
    lname varchar(30) not null,
    phone varchar(20) default '(123) 456 7890',
    photo varchar(50) default 'No photo submitted',
    street varchar(50) default '123 First St.',
    city varchar(30) default 'Toronto',
    province char(2) default 'ON',
    postcode varchar(10) default 'A1A 1A1',
    created_at timestamp default current_timestamp,
    updated_at timestamp on update current_timestamp
);
create table admins(
    id int auto_increment primary key,
```

```
email varchar(40) not null unique,  
pword varchar(255) not null  
);
```



```
Query 1 x  
use empdb;  
create table employees(  
    _id int auto_increment primary key,  
    dept varchar(30) not null,  
    paygd varchar(10) default 'none yet',  
    email varchar(40) not null unique,  
    pword varchar(255) not null,  
    fname varchar(30) not null,  
    lname varchar(30) not null,  
    phone varchar(20) default '(123) 456 7890',  
    photo varchar(50) default 'No photo submitted',  
    street varchar(50) default '123 First St.',  
    city varchar(30) default 'Toronto',  
    province char(2) default 'ON',  
    postcode varchar(10) default 'A1A 1A1',  
    created_at timestamp default current_timestamp,  
    updated_at timestamp on update current_timestamp  
);  
create table admins(  
    id int auto_increment primary key,  
    email varchar(40) not null unique,  
    pword varchar(255) not null  
);
```

After typing in the SQL statements to create the database and the tables, click on the lightning icon to execute. The database backend should now be ready.

The mysql-native library does not work in Windows 10 as of now

Disclaimer: as of this moment, the mysql-native library used for this project worked in Linux (Ubuntu). However, when I ported the project to Windows 10, I got this error:

```
ipsum ~master: building configuration "application"...
Compiling Diet HTML template index.dt...
Compiling Diet HTML template empnew.dt...
Compiling Diet HTML template emplist.dt...
Compiling Diet HTML template empedit.dt...
Compiling Diet HTML template empdelete.dt...
Compiling Diet HTML template empshow.dt...
Compiling Diet HTML template login.dt...
Linking...
ipsum.obj : error LNK2001: unresolved external symbol _D5mysql12__ModuleInfoZ
.dub\build\application-debug-windows-x86_64-dmd_v2.095.0-dirty-
7225022149FC28A6B17F7A9A8DAF2F6D\ipsum.exe : fatal error LNK1120: 1 unresolved
externals
Error: linker exited with status 1120
C:\D\dmd2\windows\bin\dmd.exe failed with exit code 1.

C:\vibeprojects\ipsum>
```

The mysql-native guys are already aware of this issue.

[Basic mysql-native dub app does not compile with dmd 2.095.0 · Issue #224 · mysql-d/mysql-native \(github.com\)](#)

But the following code works in my Ubuntu Linux system.

I won't be making a much explanations since they were mostly covered in the MongoDB version.

Preparing to access a MySQL database

Again, we are presuming that this project will only be deployed in an intranet setting and only very few people will have access and a very rare chance of more than one person using the app at the same time.

To preserve the project as it is, let us create a new project named ipsum.

```
C:\vibeprojects>dub init ipsum -t vibe.d
Package recipe format (sdl/json) [json]:
Name [ipsum]:
Description [A simple vibe.d server application.]:
Author name [rey]:
License [proprietary]:
Copyright string [Copyright © 2021, rey]:
Add dependency (leave empty to skip) []:
Successfully created an empty project in 'C:\vibeprojects\ipsum'.
Package successfully created in ipsum
```

```
C:\vibeprojects>cd ipsum
```

First, add the mysql-native library to the list of dependencies of our project. As of now, this is the dub.json file in the root directory of the project.

```
{
  "authors": [
    "rey"
  ],
  "copyright": "Copyright © 2021, rey",
  "dependencies": {
    "vibe-d": "~>0.9"
  },
  "description": "A simple vibe.d server application.",
  "license": "proprietary",
  "name": "ipsum"
}
```

It can be edited manually, just be careful with the syntax.

An alternative and safer way is to use dub to add the mysql-native library to our project.

```
C:\vibeprojects\ipsum>dub add mysql-native
Adding dependency mysql-native ~>3.0.0
```

And this is now the dub.json file, as edited by dub:

```
{
  "authors": [
    "rey"
  ],
  "copyright": "Copyright © 2021, rey",
  "dependencies": {
    "mysql-native": "~>3.0.0",
    "vibe-d": "~>0.9"
  },
  "description": "A simple vibe.d server application.",
  "license": "proprietary",
  "name": "lorem"
}
```

Copy all the files inside lorem\views* into ipsum\views. Then you must also copy the file lorem\source\empcontrol.d into ipsum\source\empcontrol.d.

Then create source\empmysql.d.

```
module empmysql;

import std.array;
import std.conv;
import mysql;

struct Employee
{
    int _id;
    string deprt;
    string paygd;
    string email;
    string pword;
    string fname;
    string lname;
    string phone;
    string photo;
    string street;
    string city;
    string province;
    string postcode;
}

struct Admin
{
    string email;
    string pword;
}
```

```
class EmployeeModel
{
    private Connection conn;

    this()
    {
        string c = "host=localhost;port=3306;user=root;pwd=password;db=empdb";
        conn = new Connection(c);
        scope(exit) conn.close;
    }

    Employee getEmployee(int id)
    {
        string sql = "select * from employees where _id=?";
        Prepared stmt = conn.prepare(sql);
        stmt.setArgs(id);
        Employee e;
        Row[] rows = conn.query(stmt).array;
        if(rows.length == 0) return e;
        return prepEmployee(rows[0]);
    }

    Employee getEmployee(string first, string last)
    {
        string sql = "select * from employees where fname=? and lname=?";
        Prepared stmt = conn.prepare(sql);
        stmt.setArgs(first, last);
        Employee e;
        Row[] rows = conn.query(stmt).array;
        if(rows.length == 0) return e;
        return prepEmployee(rows[0]);
    }

    Employee[] getEmployees()
    {
        Employee[] emps;
        Row[] rows = conn.query("select * from employees").array;
        if(rows.length) return prepEmployees(rows);
        return emps;
    }

    Employee[] getEmployees(string sql)
    {
        Row[] rows = conn.query(sql).array;
        Employee[] emps;
        if(rows.length == 0) return emps;
        return prepEmployees(rows);
    }
}
```

```
ulong addEmployee(Employee e)
{
    string sql =
        "insert into employees
        (
            dept, paygd, email, pword,
            fname, lname, phone, photo,
            street, city, province, postcode
        )
        values(?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)";
    Prepared stmt = conn.prepare(sql);
    stmt.setArgs
    (
        to!string(e.deprt),
        to!string(e.paygd),
        to!string(e.email),
        to!string(e.pword),
        to!string(e.fname),
        to!string(e.lname),
        to!string(e.phone),
        to!string(e.photo),
        to!string(e.street),
        to!string(e.city),
        to!string(e.province),
        to!string(e.postcode)
    );
    return conn.exec(stmt);
}

ulong editEmployee(Employee e)
{
    string sql = "update employees set
        dept=? , paygd=? , email=? , pword=? ,
        fname=? , lname=? , phone=? , photo=? ,
        street=? , city=? , province=? , postcode=?
        where _id=?";
    Prepared stmt = conn.prepare(sql);
    stmt.setArgs
    (
        to!string(e.deprt),
        to!string(e.paygd),
        to!string(e.email),
        to!string(e.pword),
        to!string(e.fname),
        to!string(e.lname),
        to!string(e.phone),
        to!string(e.photo),
        to!string(e.street),
        to!string(e.city),
        to!string(e.province),
```

```

        to!string(e.postcode),
        to!int(to!string(e._id))
    );
    return conn.exec(stmt);
}

ulong deleteEmployee(int id)
{
    string sql = "delete from employees where _id=?";
    Prepared stmt = conn.prepare(sql);
    stmt.setArgs(id);
    return conn.exec(stmt);
}

bool getAdmin(string email, string pword)
{
    string sql = "select * from admins where email=? and pword=?";
    Prepared stmt = conn.prepare(sql);
    stmt.setArgs(email, pword);
    Row[] rows = conn.query(stmt).array;
    if(rows.length) return true;
    return false;
}

Admin[] getAdmins()
{
    Admin[] admins;
    Row[] rows = conn.query("select email, pword from admins").array;
    foreach (Row row; rows)
    {
        Admin a;
        a.email = to!string(row[0]);
        a.pword = to!string(row[1]);
        admins ~= a;
    }
    return admins;
}

void encryptAdminPassword(string email, string passw)
{
    string sql = "update admins set pword=? where email=?";
    Prepared stmt = conn.prepare(sql);
    stmt.setArgs(passw, email);
    conn.exec(stmt);
}

Employee[] prepEmployees(Row[] rows)
{
    Employee[] emps;
    foreach(Row row; rows)

```

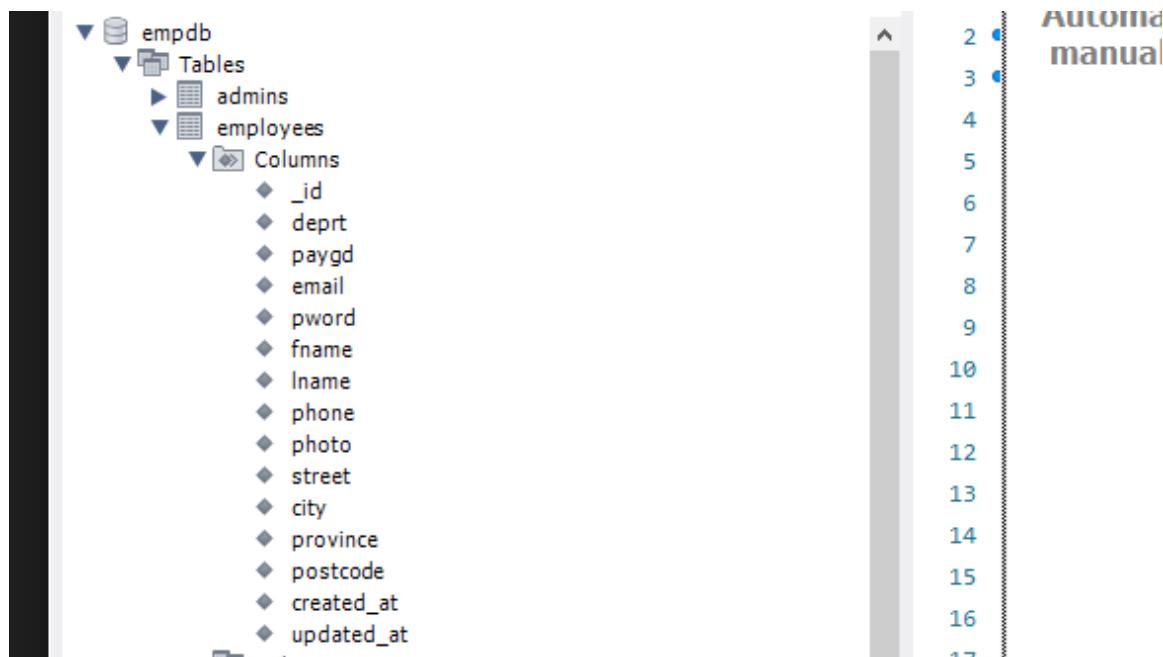
```

    {
        Employee e = prepEmployee(row);
        emps ~= e;
    }
    return emps;
}

Employee prepEmployee(Row row)
{
    Employee e;
    e._id = to!int(to!string(row[0]));
    e.deprt = to!string(row[1]);
    e.paygd = to!string(row[2]);
    e.email = to!string(row[3]);
    e.pword = to!string(row[4]);
    e.fname = to!string(row[5]);
    e.lname = to!string(row[6]);
    e.phone = to!string(row[7]);
    e.photo = to!string(row[8]);
    e.street = to!string(row[9]);
    e.city = to!string(row[10]);
    e.province = to!string(row[11]);
    e.postcode = to!string(row[12]);
    return e;
}
}

```

In prepEmployee() function, you should make the order of fields in your code conform to the order of fields in your MySQL table.



Sometimes, MySQL does not save the fields in the way you created the table.

Here is the full source\empcontrol.d.

```
module empcontrol;

import vibe.vibe;
import vibe.http.auth.digest_auth;
import std.file;
import std.path;
import std.algorithm;
import empmysql;

struct User
{
    bool loggedIn;
    string email;
}

class EmployeeController
{
    private EmployeeModel empModel;
    private SessionVar!(User, "user") m_user;
    string realm = "Lorem Ipsum Company";
    string[] deps =
    [
        "Management and Admin",
        "Accounting and Finance",
        "Production",
        "Maintenance",
        "Shipping and Receiving",
        "Purchasing and Supplies",
        "IT Services",
        "Human Resources",
        "Marketing"
    ];
    string[][][] provs =
    [
        ["AB", "Alberta"],
        ["BC", "British Columbia"],
        ["MB", "Manitoba"],
        ["NB", "New Brunswick"],
        ["NL", "Newfoundland and Labrador"],
        ["NS", "Nova Scotia"],
        ["NT", "Northwest Territories"],
        ["NU", "Nunavut"],
        ["ON", "Ontario"],
        ["PE", "Prince Edward Island"],
    ];
}
```

```

        ["QC", "Quebec"],
        ["SK", "Saskatchewan"],
        ["YT", "Yukon Territory"]
    ];

this()
{
    empModel = new EmployeeModel;
}

void index()
{
    m_user = User.init;
    terminateSession;
    render!("index.dt");
}

@auth
void getNewEmployee(string _authUser, string _error = null)
{
    string error = _error;
    render!("empnew.dt", deps, provs, error);
}

@errorDisplay!getNewEmployee
@auth
void postNewEmployee(string _authUser, Employee e)
{
    string photopath = "No photo submitted";
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.addEmployee(e);
}

```

```

        redirect("list_employees");
    }

@auth
void getListEmployees(string _authUser, string _error = null)
{
    string error = _error;
    Employee[] emps = empModel.getEmployees();
    render!("emplist.dt", emps, error);
}

@auth
void getEditEmployee(string _authUser, int id, string _error = null)
{
    string error = _error;
    Employee e = empModel.getEmployee(id);
    render!("empedit.dt", e, deps, provs, error);
}

@errorDisplay!getEditEmployee
@auth
void postEditEmployee(string _authUser, Employee e)
{
    string photopath = e.photo;
    auto pic = "picture" in request.files;
    if(pic !is null)
    {
        string ext = extension(pic.filename.name);
        string[] exts = [".jpg", ".jpeg", ".png", ".gif"];
        if(canFind(exts, ext))
        {
            photopath = "uploads/photos/" ~ e.fname ~ "_" ~ e.lname ~ ext;
            string dir = "./public/uploads/photos/";
            mkdirRecurse(dir);
            string fullpath = dir ~ e.fname ~ "_" ~ e.lname ~ ext;
            try moveFile(pic.tempPath, NativePath(fullpath));
            catch (Exception ex) copyFile(pic.tempPath, NativePath(fullpath));
        }
    }
    e.photo = photopath;
    if(e.phone.length == 0) e.phone = "(123) 456 7890";
    if(e.paygd.length == 0) e.paygd = "none yet";
    if(e.postcode.length == 0) e.postcode = "A1A 1A1";
    e.pword = createDigestPassword(realm, e.email, e.pword);
    empModel.editEmployee(e);
    redirect("list_employees");
}

@auth

```

```

void getDeleteEmployee(string _authUser, int id)
{
    Employee e = empModel.getEmployee(id);
    render!("empdelete.dt", e);
}

@auth
void postDeleteEmployee(string _authUser, int id)
{
    empModel.deleteEmployee(id);
    redirect("list_employees");
}

@errorDisplay!getListEmployees
@auth
void getShowEmployee(string _authUser, string fname, string lname)
{
    Employee e = empModel.getEmployee(fname, lname);
    enforce(e != Employee.init, "Employee not found!");
    render!("empshow.dt", e);
}

void getLogin(string _error = null)
{
    string error = _error;
    render!("login.dt", error);
}

@errorDisplay!getLogin
void postLogin(string email, string password)
{
    auto pword = createDigestPassword(realm, email, password);
    bool isAdmin = empModel.getAdmin(email, pword);
    enforce(isAdmin, "Email and password combination not found!");
    User user = m_user;
    user.loggedIn = true;
    user.email = email;
    m_user = user;
    redirect("list_employees");
}

void getEncryptPasswords()
{
    Admin[] admins = empModel.getAdmins;
    foreach (Admin a; admins)
    {
        auto password = createDigestPassword(realm, a.email, a.pword);
        empModel.encryptAdminPassword(a.email, password);
    }
    redirect("list_employees");
}

```

```

}

private enum auth = before!ensureAuth("_authUser");

private string ensureAuth(HTTPRequest req, HTTPServerResponse res)
{
    if(!m_user.loggedIn) redirect("login");
    return m_user.email;
}
mixin PrivateAccessProxy;
}

```

The views\empdelete.dt.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2.brown Are you sure you want to delete this record?
        form.form-grid(method="post", action="delete_employee")
            div.form-grid-column
                | Department:<br />
                div.form-grid-column-field #{e.deprt}
            div.form-grid-column
                | Salary grade:<br />
                div.form-grid-column-field #{e.paygd}
            div.form-grid-column
                | Email address:<br />
                div.form-grid-column-field #{e.email}
            div.form-grid-column
                | Password:<br />
                div.form-grid-column-field *****
            div.form-grid-column
                | First name:<br />
                div.form-grid-column-field #{e.fname}
            div.form-grid-column
                | Last name:<br />
                div.form-grid-column-field #{e.lname}
            div.form-grid-column
                | Phone:<br />
                div.form-grid-column-field #{e.phone}
            div.form-grid-column
                | Street address (without city):<br />
                div.form-grid-column-field #{e.street}
            div.form-grid-column
                | City:<br />
                div.form-grid-column-field #{e.city}
            div.form-grid-column
                | Province:<br />

```

```

    div.form-grid-column-field #{e.province}
    div.form-grid-column
      | Postal code:<br />
      div.form-grid-column-field #{e.postcode}
    div.form-grid-column
      | Profile picture:<br />
      img(src="#{e.photo}", height="100px")
      input(type="hidden", name="id", value="#{e._id}")
    div.form-grid-column
      a(href="list_employees")
        button.form-grid-column-button(type="button") No
    div.form-grid-column
      input.form-grid-column-button(type="submit", value="Yes")

```

The views\empedit.dt.

```

extends layout
block maincontent
  include cssformgrid.dt
  div.form-grid-wrapper
    h2 Edit employee details
    form.form-grid(method="post", action="edit_employee", enctype="multipart/form-data")
      div.form-grid-column Department<br />
        select#depid.form-grid-column-input(name="e_deprt", value="#{e.deprt}")
          - foreach(dep; deps)
            - if(dep == e.deprt)
              option(value="#{dep}", selected) #{dep}
            - else
              option(value="#{dep}") #{dep}
      div.form-grid-column Salary grade<br />
        input.form-grid-column-
      input(type="text", name="e_paygd", value="#{e.paygd}")
      div.form-grid-column Email address<br />
        input.form-grid-column-
      input(type="email", name="e_email", value="#{e.email}")
      div.form-grid-column Password<br />
        input.form-grid-column-
      input(type="password", name="e_pword", value="#{e.pword}")
      div.form-grid-column First name<br />
        input.form-grid-column-
      input(type="text", name="e_fname", value="#{e.fname}")
      div.form-grid-column Last name<br />
        input.form-grid-column-
      input(type="text", name="e_lname", value="#{e.lname}")
      div.form-grid-column Phone<br />
        input.form-grid-column-
      input(type="text", name="e_phone", value="#{e.phone}")
      div.form-grid-column Street address (without city)<br />

```

```

    input.form-grid-column-


```

The views\emplist.dt.

```

extends layout
block maincontent
    include csstable.dt
    div.table-wrapper
        table
            tr
                td.no-border(colspan="11")
                - if(error)
                    span.error #{error}<br /><br />
            tr
                th Department
                th Salary grade
                th First name
                th Last name
                th Phone number
                th Email address

```

```

th Street address
th City
th Province
th PostCode
th Action
- foreach(e; emps)
tr
td #{e.deprt}
td #{e.paygd}
td #{e.fname}
td #{e.lname}
td #{e.phone}
td #{e.email}
td #{e.street}
td #{e.city}
td #{e.province}
td #{e.postcode}
td &nbs;
form.form-hidden(method="get", action="edit_employee")
input(type="hidden", name="id", value="#{e._id}")
input(type="image", src="images/pen.ico")
| &nbs;
form.form-hidden(method="get", action="delete_employee")
input(type="hidden", name="id", value="#{e._id}")
input(type="image", src="images/trash.ico")
| &nbs;

```

The views\empnew.dt.

```

extends layout
block maincontent
  include cssformgrid.dt
  div.form-grid-wrapper
    h2 New employee details
    form.form-grid(method="post", action="new_employee", enctype="multipart/form-data")
      div.form-grid-column Department<br />
        select#dept.form-grid-column-input(name="e_dept")
          - foreach(dep; deps)
            option(value="#{dep}") #{dep}
      div.form-grid-column Salary grade<br />
        input.form-grid-column-
input(type="text", name="e_paygd", placeholder="Salary grade")
      div.form-grid-column Email address<br />
        input.form-grid-column-
input(type="email", name="e_email", placeholder="email@address.com", required)
      div.form-grid-column Password<br />

```

```

        input.form-grid-column-
input(type="password", name="e_pword", placeholder="password", required)
    div.form-grid-column First name<br />
        input.form-grid-column-
input(type="text", name="e_fname", placeholder="First name", required)
    div.form-grid-column Last name<br />
        input.form-grid-column-
input(type="text", name="e_lname", placeholder="Last name", required)
    div.form-grid-column Phone<br />
        input.form-grid-column-
input(type="text", name="e_phone", placeholder="Phone number")
    div.form-grid-column Street address (without city)<br />
        input.form-grid-column-
input(type="text", name="e_street", placeholder="Street address", required)
    div.form-grid-column City<br />
        input.form-grid-column-
input(type="text", name="e_city", placeholder="City", required)
    div.form-grid-column Province<br />
        select#province.form-grid-column-input(name="e_province")
            - foreach(prov; provs)
                option(value="#{prov[0]}") #{prov[1]}
    div.form-grid-column Postal code<br />
        input.form-grid-column-
input(type="text", name="e_postcode", placeholder="A1A 1A1")
    div.form-grid-column ID Picture<br />
        input.form-grid-column-input(type="file", name="picture")
    input(type="hidden", name="e_photo")
    input(type="hidden", name="e_id", value="0")
    div.form-grid-column
        br
        input.form-grid-column-button(type="reset", value="Clear")
    div.form-grid-column
        br
        input.form-grid-column-button(type="submit", value="Submit")
- if(error)
    div.error #{error}

```

And here is views\empshow.dt.

```

extends layout
block maincontent
    include cssformgrid.dt
    div.form-grid-wrapper
        h2 Employee details
        form.form-grid(method="get", action="edit_employee")
            div.form-grid-column
                | Department<br />
                div.form-grid-column-field #{e.deprt}
            div.form-grid-column

```

```

| Salary grade<br />


```

Compile, run and refresh the browser. If you see the same app that we did with MongoDB, you did good.

We are done!

Hello, main() again!

D is a case-sensitive language, just like the other C-derived languages.

To create a non-Vibe.d project in D using dub, what we type is

```
dub init hello
```

That's it. No -t vibe.d anymore.

Compiling and running is the same, just

```
dub
```

So let's create the hello project.

```
C:\dprojects>dub init hello
Package recipe format (sdl/json) [json]:
Name [hello]:
Description [A minimal D application.]:
Author name [rey]:
License [proprietary]:
Copyright string [Copyright © 2021, rey]:
Add dependency (leave empty to skip) []:
Successfully created an empty project in 'C:\dprojects\hello'.
Package successfully created in hello
```

```
C:\dprojects>cd hello
```

```
C:\dprojects\hello>dub
Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.
hello ~master: building configuration "application"...
Linking...
Running .\hello.exe
Edit source/app.d to start your project.
```

```
C:\dprojects\hello>
```

No 'Hello, world' greeting? That's unheard of. Let's rectify that.

Edit source\app.d and change the offending text.

```
import std.stdio;
void main()
```

```
{  
    writeln("Hello, world!");  
}
```

C:\dprojects\hello>**dub**

Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.

hello ~master: building configuration "application"...

Linking...

Running .\hello.exe

Hello, world!

C:\dprojects\hello>

There.

You see that the code is similar to C or C++. That's because Walter Bright (later aided by Andrei Alexandrescu, then a host of others) wanted to create a better C or C++ minus the extra complexities of C++.

We need to import std.stdio because the writeln() function is in that library, along with other input/output functions. Some of the I/O functions in std.stdio are

```
write  
writeln  
writef  
writelnf  
readf  
readln  
stdin  
stdout
```

The main() function is the entry point for D programs. Even if the project is composed of several files, only one main() function is required as the entry point and it determines the order of execution of the whole program.

The main() function can also accept arguments.

```
import std.stdio;  
  
void main(string[] args)  
{  
    foreach(arg; args) writeln(arg);  
}
```

Compile the project.

```
C:\dprojects\hello>dub
Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.
hello ~master: building configuration "application"...
Linking...
Running .\hello.exe
.\hello.exe
```

Then run the resulting executable with some arguments.

```
C:\dprojects\hello>hello.exe how are you doing today?
hello.exe
how
are
you
doing
today?
```

```
C:\dprojects\hello>
```

The foreach() construct is heaven-sent as it makes looping through an iterable data structure so convenient.

Although we can still use the C-style for() loop.

```
import std.stdio;
import std.conv;

void main(string[] args)
{
    for(int i = 0; i < args.length; i++) writeln(to!string(i) ~ ":" ~ args[i]);
}
```

We have to import std.conv because the conversion functions are there, such as to!string().

Of course we should be using curly brackets inside the loop to signify the block of code following convention, but sometimes we are just lazy.

```
import std.stdio;

void main(string[] args)
{
    for(int i = 0; i < args.length; i++)
    {
        writeln(to!string(i) ~ ":" ~ args[i]);
    }
}
```

```
}
```

```
C:\dprojects\hello>dub
Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.
hello ~master: building configuration "application"...
Linking...
Running .\hello.exe
0: .\hello.exe
```

```
C:\dprojects\hello>hello.exe how are you doing today?
0: hello.exe
1: how
2: are
3: you
4: doing
5: today?
```

We can also use the formatted way of printing to stdout by using `writef()` or `writeln()`. This time we don't need the conversion function, so we don't have to import `std.conv`.

```
import std.stdio;

void main(string[] args)
{
    for(int i = 0; i < args.length; i++)
    {
        writeln("%d: %s", i, args[i]);
    }
}
```

```
C:\dprojects\hello>dub
Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.
hello ~master: building configuration "application"...
Linking...
Running .\hello.exe
0: .\hello.exe
```

```
C:\dprojects\hello>hello.exe how are you doing today?
0: hello.exe
1: how
2: are
3: you
4: doing
5: today?
```

```
C:\dprojects\hello>
```

The main() function can also return a value, called the exit status, although a non-zero value means an error occurred.

```
import std.stdio;

int main(string[] args)
{
    writeln("Hello, how are you?");
    return 0;
}
```

```
C:\dprojects\hello>dub
```

Performing "debug" build using C:\D\dmd2\windows\bin\dmd.exe for x86_64.

hello ~master: building configuration "application"...

Linking...

Running .\hello.exe

Hello, how are you?

```
C:\dprojects\hello>
```

Here's another example.

```
import std.stdio;

int main(string[] args)
{
    writeln("Hello, how are you?");
    return 0;
}
```

This time, we indicated an int return value as well as an array of strings as parameter. We did not use the argument args within the function though.

```
    writeln("Hello, how are you?");
```

The writeln() and writefln() functions write to stdout, which is the screen. It is included in the std.stdio module, that's why we imported std.stdio.

Instead of using dub, we can simply call the DMD compiler directly if we are writing a simple program that doesn't need the added complexity of a project structure. To use the DMD compiler, you should be in the same folder as the source file you want to compile.

```
C:\dprojects\hello>cd source
```

```
C:\dprojects\hello\source>dir  
Volume in drive C has no label.  
Volume Serial Number is D4B4-80DE
```

Directory of C:\dprojects\hello\source

```
2021-02-18 12:03 PM <DIR> .  
2021-02-18 12:03 PM <DIR> ..  
2021-02-18 10:36 PM 91 app.d  
    1 File(s)      91 bytes  
2 Dir(s) 407,335,723,008 bytes free
```

```
C:\dprojects\hello\source>dmd app.d
```

```
C:\dprojects\hello\source>dir  
Volume in drive C has no label.  
Volume Serial Number is D4B4-80DE
```

Directory of C:\dprojects\hello\source

```
2021-02-18 10:58 PM <DIR> .  
2021-02-18 10:58 PM <DIR> ..  
2021-02-18 10:36 PM 91 app.d  
2021-02-18 10:58 PM 267,292 app.exe  
2021-02-18 10:58 PM 8,868 app.obj  
    3 File(s) 276,251 bytes  
2 Dir(s) 407,335,362,560 bytes free
```

```
C:\dprojects\hello\source>app.exe  
Hello, how are you?
```

```
C:\dprojects\hello\source>
```

D keywords

Here are the D keywords to avoid when declaring variables, constants and functions.

abstract	else	macro	template
alias	enum	mixin	this
align	export	module	throw
asm	extern		true
assert		new	try
auto	false	nothrow	typeid
	final	null	typeof
body	finally		
bool	float	out	ubyte
break	for	override	ucent
byte	foreach		uint
	foreach_reverse	package	ulong
case	function	pragma	union
cast		private	unittest
catch	goto	protected	ushort
cdouble		public	
cent	idouble	pure	version
cfloat	if	real	void
char	ifloat	ref	
class	immutable	return	wchar
const	import		while
continue	in	scope	with
creal	inout	shared	
	int	short	__FILE__
dchar	interface	static	__FILE_FULL_PATH__
debug	invariant	struct	__MODULE__
default	ireal	super	__LINE__
delegate	is	switch	__FUNCTION__
delete		synchronized	__PRETTY_FUNCTION__
deprecated	lazy		
do	long		
double			__gshared
			__traits
			__vector
			__parameters

Special tokens

<u>__DATE__</u>	string literal of date of compilation "mmm dd yyyy"
<u>__EOF__</u>	tells the scanner to ignore everything after this token
<u>__TIME__</u>	string literal of time of compilation "hh:mm:ss"
<u>__TIMESTAMP__</u>	string literal of date and time of compilation "www mmm dd hh:mm:ss yyyy"
<u>__VENDOR__</u>	Compiler vendor string
<u>__VERSION__</u>	Compiler version as an integer

Data types

Here is the list of all the basic data types.

Keyword	Default Initializer (.init)	Description
<code>void</code>	-	no type
<code>bool</code>	<code>false</code>	boolean value
<code>byte</code>	<code>0</code>	signed 8 bits
<code>ubyte</code>	<code>0u</code>	unsigned 8 bits
<code>short</code>	<code>0</code>	signed 16 bits
<code>ushort</code>	<code>0u</code>	unsigned 16 bits
<code>int</code>	<code>0</code>	signed 32 bits
<code>uint</code>	<code>0u</code>	unsigned 32 bits
<code>long</code>	<code>0L</code>	signed 64 bits
<code>ulong</code>	<code>0uL</code>	unsigned 64 bits
<code>cent</code>	<code>0</code>	signed 128 bits (reserved for future use)
<code>ucent</code>	<code>0u</code>	unsigned 128 bits (reserved for future use)
<code>float</code>	<code>float.nan</code>	32 bit floating point
<code>double</code>	<code>double.nan</code>	64 bit floating point
<code>real</code>	<code>real.nan</code>	largest FP size implemented in hardware <i>(Implementation Note: 80 bits for x86 CPUs or <code>double</code> size, whichever is larger)</i>
<code>ifloat</code>	<code>float.nan*1.0i</code>	imaginary float
<code>idouble</code>	<code>double.nan*1.0i</code>	imaginary double
<code>ireal</code>	<code>real.nan*1.0i</code>	imaginary real
<code>cfloat</code>	<code>float.nan+float.nan*1.0i</code>	a complex number of two float values
<code>cdouble</code>	<code>double.nan+double.nan*1.0i</code>	complex double
<code>creal</code>	<code>real.nan+real.nan*1.0i</code>	complex real
<code>char</code>	<code>'xFF'</code>	unsigned 8 bit (UTF-8 code unit)
<code>wchar</code>	<code>'uFFFF'</code>	unsigned 16 bit (UTF-16 code unit)
<code>dchar</code>	<code>'U0000FFFF'</code>	unsigned 32 bit (UTF-32 code unit)

Operators, expressions and statements

Variables represent values. If you want to be more precise, a variable represents a region in memory where a value is stored. (That was *technical!*)

You must be familiar with this syntax if you came from the C-related languages.

```
int a = 0;  
bool isReady = false;  
double netIncome = grossIncome - deductions;  
float price = itemCost + freight * duties;
```

To declare a variable, the data type comes first before the variable name.

```
int a = 0;
```

In this case, a is declared an integer and was assigned an initial value of 0;

```
bool isReady = false;
```

In this case, isReady is declared as a Boolean variable with an initial value of false.

Operations on variables are expressions. When you assign the result of an expression to something, that becomes a statement. Yet simply declaring a variable or a user-defined data type is already a statement. Go figure.

```
a == 0;           // an expression (double equals)  
x + b -10;      // an expression  
cost * 1.2 / 3 * items; // an expression  
nonZero = a >= 1; // a statement  
y = x + b -10; // a statement  
total = cost * 1.2 / 3 * items; // a statement  
int a;           // a statement  
struct x { int a; int b; } // a statement
```

Since you are interested in Vibe.d, you must be familiar with these already.

=	assignment operator
==	equality operator
+	addition operator
*	multiplication operator
-	subtraction operator
/	division operator
>=	relational (comparison) operator