`mix phx.gen.live`

Kairen, Airuddinsah, Mark, Nathaniel, Elvin

mix phx.gen.live Accounts User users name:string age:integer

Mix	Command related to phoenix and elixir
phx.gen.live	Pnx Liveview generator. Build all files for create, read, update,delete operations
Accounts	Context name
User	Schema name (elixir module representing db table)
users	Database table data
Name:string age:integer	Define field/column

mix ecto.migrate

Apply database changes in the actual database (postgres)

	id [PK] bigint	name character varying (255)	age integer	inserted_at timestamp without time zone	updated_at timestamp without time zone
1	1	Alice	24	2025-07-02 01:42:39	2025-07-02 01:42:39
2	2	Bob	26	2025-07-02 01:42:49	2025-07-02 01:42:49
3	7	Jake	126	2025-07-02 05:45:47	2025-07-02 05:45:47

Get vs Live

```
get "/", PageController, :home

live "/users", UserLive.Index, :index
live "/users/new", UserLive.Index, :new
live "/users/:id/edit", UserLive.Index, :edit
```

route.ex

Get is used for static pages. Every time changes happen, page refresh is required

- E-commerce apps

Live keep live connection between browser and server. Used for real time stuffs.

- Live chat during livestream,
- Filters,
- Live interactive dashboard - polling result

You want	Use ‘get’	Use ‘live’
Basic simple web page	Yes	No
Dynamic edit	No	Yes
Real time form validation	Require JS	Built in
Real time feature	Hard to do	Built in with Liveview


```
def create_user(attrs \\ %{}) do
  %User{}
  |> User.changeset(attrs)
  |> Repo.insert()
end
```

accounts.ex

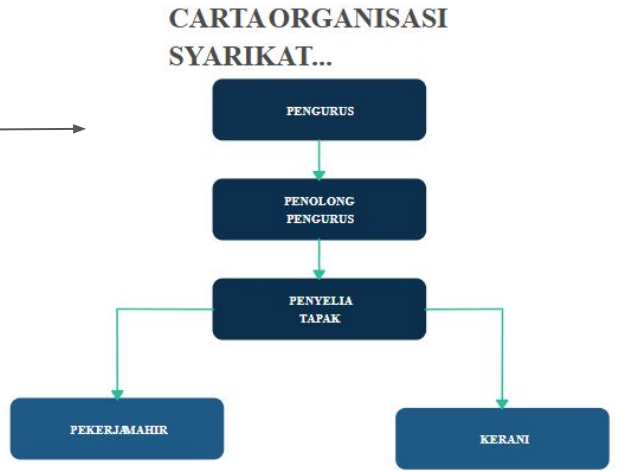
```
def changeset(user, attrs) do
  user
  |> cast(attrs, [:name, :age])
  |> validate_required([:name, :age])
end
```

user.ex

Context

- A module that stored schema needed for the module

Example




```elixir
defmodule TestApp.Accounts.User do
  use Ecto.Schema
  import Ecto.Changeset

  schema "users" do
    field :name, :string
    field :age, :integer

    timestamps(type: :utc_datetime)
  end

  @doc false
  def changeset(user, attrs) do
    user
    |> cast(attrs, [:name, :age])
    |> validate_required([:name, :age])
  end
end
```

- TestApp is the module, hence it requires user information such as name and age.
- timestamps record information on “When was it added” and “When was it last updated”
- On changeset, it makes that input “name and age” does not show. Also, both is compulsory to be filled.

mix phx.gen.live: Satu *Mix task* dalam Phoenix yang digunakan untuk menjana struktur *LiveView* secara automatik bagi sesuatu sumber (*resource*) yang menyokong operasi CRUD (*Create, Read, Update, Delete*).

- mix phx.gen.live Accounts User users name:string age:integer
- mix ecto.migrate
- iex -S mix phx.server

routes.ex

```
scope "/", MyAppWeb do
  pipe_through :browser

  get "/", PageController, :index

  live "/users", UserLive.Index, :index
  live "/users/new", UserLive.Index, :new
  live "/users/:id/edit", UserLive.Index, :edit


  live "/users/:id", UserLive.Show, :show
  live "/users/:id/show/edit", UserLive.Show, :edit
```

Senarai pengguna

Tambah pengguna baru

Edit pengguna (senarai)

Lihat pengguna tertentu

Edit pengguna (dari paparan)

```
.ex M   20250702061444_create_users.exs U   use
app > accounts > user.ex > {} MyApp.Accounts.User
defmodule MyApp.Accounts.User do
  use Ecto.Schema
  import Ecto.Changeset

  schema "users" do
    field :name, :string
    field :age, :integer
    field :phone, :integer

    timestamps(type: :utc_datetime)
  end

  @doc false
  def changeset(user, attrs) do
    user
    |> cast(attrs, [:name, :age, :phone])
    |> validate_required([:name, :age, :phone])
  end
end
```

Mewakili data seorang pengguna

mengaktifkan ciri-ciri khas dari `Ecto`

Modul ini mewakili jadual bernama `users` dalam pangkalan data.

Ini digunakan bila anda mahu cipta atau ubah data pengguna

```
r.ex M   20250702032855_create_messages.exs U
po > migrations > 20250702061444_create_users.exs > ...
defmodule MyApp.Repo.Migrations.CreateUsers do
  use Ecto.Migration

  def change do
    create table(:users) do
      add :name, :string
      add :age, :integer
      add :phone, :integer

      timestamps(type: :utc_datetime)
    end
  end
end
```

Ini ialah modul migrasi untuk mencipta jadual baru bernama `users` dalam pangkalan data.

Aktifkan semua fungsi khas untuk migrasi, seperti `create`, `add`, dan `drop`.

tempat anda tulis perubahan struktur pangkalan data.

Cipta **jadual baru** bernama `users`.

Repo

- A module that interface to database. Involve CRUD.

Data Output | Messages | Notifications

	id [PK] bigint	name character varying (255)	age integer	inserted_at timestamp without time zone	updated_at timestamp without time zone
1	1	Van	25	2025-07-02 03:47:19	2025-07-02 03:47:19

- Any changes on the user information will be recorded to the database as shown above

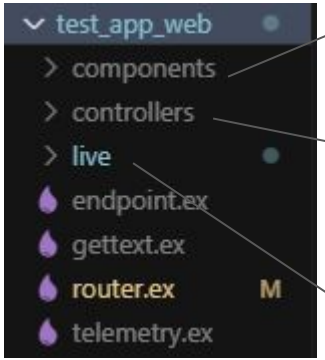
Example

```elixir
defmodule TestApp.Accounts do
  def get_user!(id), do: Repo.get!(User, id)

  @doc """
  Creates a user.

  ## Examples

      iex> create_user(%{field: value})
      {:ok, %User{}}

      iex> create_user(%{field: bad_value})
      {:error, %Ecto.Changeset{}}

  """
  def create_user(attrs \\ %{}) do
    %User{}
    |> User.changeset(attrs)
    |> Repo.insert()
  end

  @doc """
  Updates a user.

  ## Examples

      iex> update_user(user, %{field: new_value})
      {:ok, %User{}}

      iex> update_user(user, %{field: bad_value})
      {:error, %Ecto.Changeset{}}

  """
  def update_user(%User{} = user, attrs) do
    user
    |> User.changeset(attrs)
    |> Repo.update()
  end
```

Web namespace

- Handles web-related codes such as Components, Controllers, LiveViews etc

Example



Allow to reuse an interface elements

A part of MVC architecture

Store live components that contain full LiveView

endpoint.ex	• The gatekeeper between http to app Exp: Chrome > endpoint > Router > LiveView/Controller > HTML > Chrome
gettext.ex	Translation generator that translate what user type setted in the code
router.ex	Act like a guide that show you different part of website based on URL
telemetry.ex	Helps you understand how well your app is performing and whether it's working as expected.

Comparison

Controller-Base Page Flow

✅ Flow for Controller-Based Page (Traditional MVC)

scss Copy Edit

```
Browser (Chrome)
   ↓ HTTP Request
[ Endpoint ]  ← gatekeeper (sessions, cookies, parsing, etc.)
   ↓
[ Router ]    ← matches URL (like `/about`)
   ↓
[ Controller ] ← handles logic, fetches data
   ↓
[ View ]       ← renders a template (like `about.html.heex`)
   ↓
[ HTML Response ]
   ↓
Browser (Chrome) shows rendered page
```

LiveView Page Flow

⚡ Flow for a LiveView Page (Real-Time, Interactive)

This flow is **slightly different** because LiveView uses a **WebSocket** after the first render:

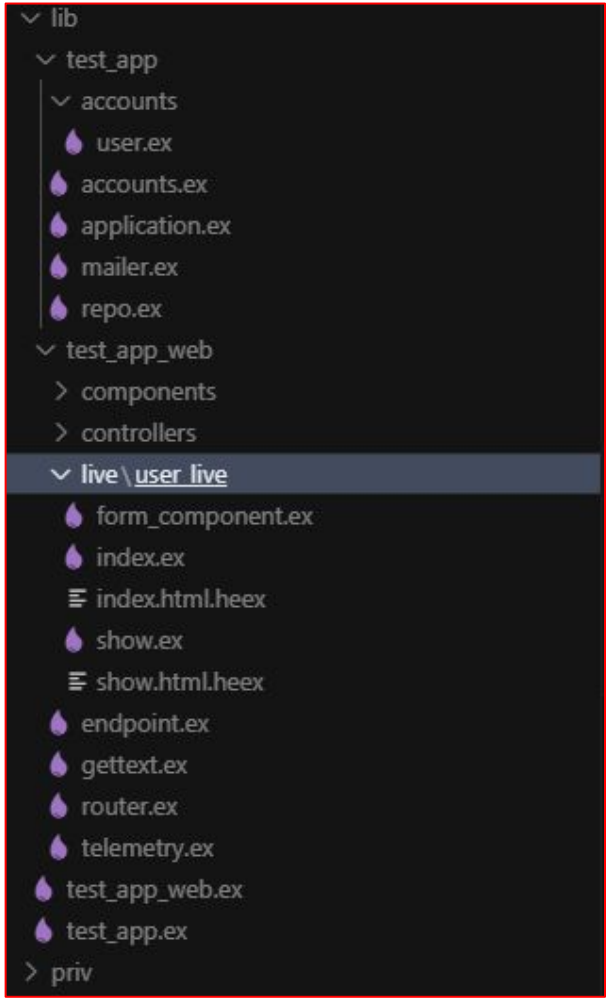
scss Copy Edit

```
Browser (Chrome)
   ↓ HTTP Request
[ Endpoint ]
   ↓
[ Router ]
   ↓
[ LiveView ] (mounts real-time socket connection)
   ↓
[ Template rendered with assigns ]
   ↓
HTML (initial render) sent to Chrome
   ↓
💬 WebSocket opens (for live interaction)
   ↕
LiveView (stays connected)
   ↔ user clicks, types, events
   ↔ LiveView sends only changed parts of page
```

Kairen Deiwien

Overall, this generator will add the following files:

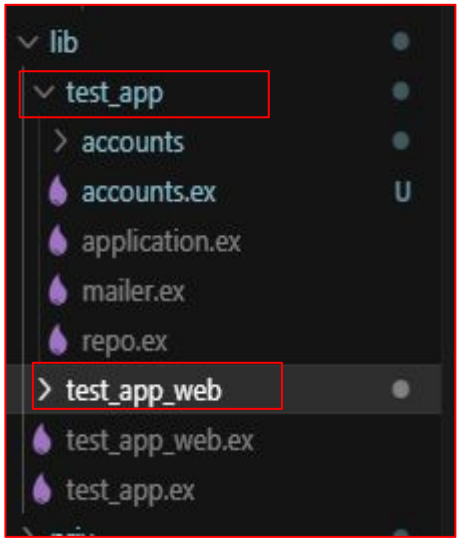
- a context module in `lib/app/accounts.ex` for the accounts API
- a schema in `lib/app/accounts/user.ex`, with a `users` table
- a LiveView in `lib/app_web/live/user_live/show.ex`
- a LiveView in `lib/app_web/live/user_live/index.ex`
- a LiveComponent in `lib/app_web/live/user_live/form_component.ex`
- a helpers module in `lib/app_web/live/live_helpers.ex` with a modal



Kairen Deiwien

File	Purpose
`accounts.ex`	Business logic (user creation, update,read. delete.)
`user.ex`	Schema and changeset (to defines what a user is / validation)
`index.ex`	Lists users and opens form modal
`show.ex`	Displays one user, with edit modal
`form_component.ex`	Shared form logic (both Creating and Editing users)
`live_helpers.ex`	UI modal logic (for `live_modal`)

Kairen Deiwien



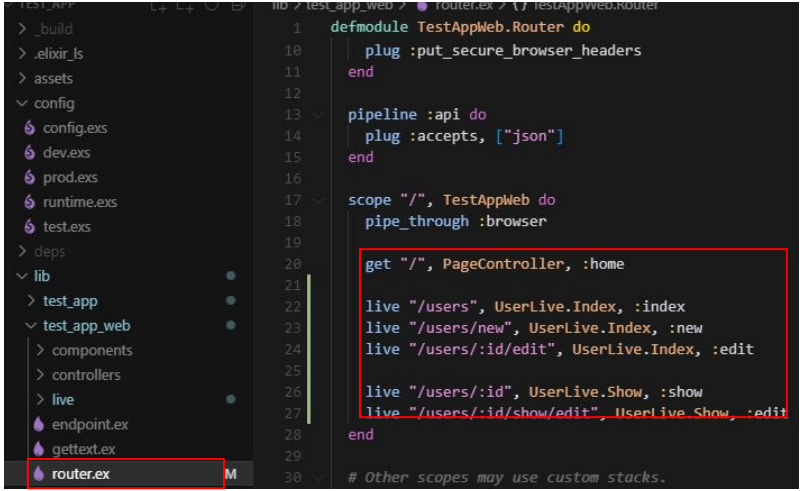
Aspects	Test_app File	Test_app Web File
Purpose	Core business logic and data access	Web interface (HTTP,html, liveview,routing)
Typical modules	`Accounts`, `User`, `MyApp.Repo`	`UserController`, `UserLive.Index`, `UserView`
concerns	"What the app does"	"How the user sees/interacts with it"
Main contain	Contexts, schemas, changesets, Repo	Controllers, LiveViews, templates, components, router
Ecto usage	Yes - queries, validation, db operations	Rare - mostly calls context functions like Accounts.list_users()
User interaction	No direct interaction	Handles events, clicks, form submits

`Summary:`

`lib/my_app/` = **Brains**: "This is how users are stored, validated, and saved."

`lib/my_app_web/` = **Face**: "Here's the HTML page and buttons the user clicks."

Kairen Deiwien



Liveview route

URL	Module	Action	Purpose
`/users`	`UserLive.Index`	`:index`	Show all users
`/users/new`	`UserLive.Index`	`:new`	Form to add user
`/users/:id/edit`	`UserLive.Index`	`:edit`	Edit user (from list)
`/users/:id`	`UserLive.Show`	`:show`	Show user detail
`/users/:id/show/edit`	`UserLive.Show`	`:edit`	Edit from detail page

What is `.heex`?

- `.heex` stands for HTML Embedded Elixir (HEEx).

- It is a file format used for building dynamic HTML pages in Phoenix Framework.

- Allows you to write HTML + Elixir code together.

- Supports LiveView features (real-time updates).

What is `home.html.heex`?

- `home.html.heex` is a template file.

- It defines the visual layout and content of your website's homepage.

- Located in: lib/your_app_web/controllers/page/home.html.heex

- It is rendered by: PageController.home/2 in your router and controller files.

Function of `home.html.heex`

- Acts as the user-facing page for the home route (`/`).

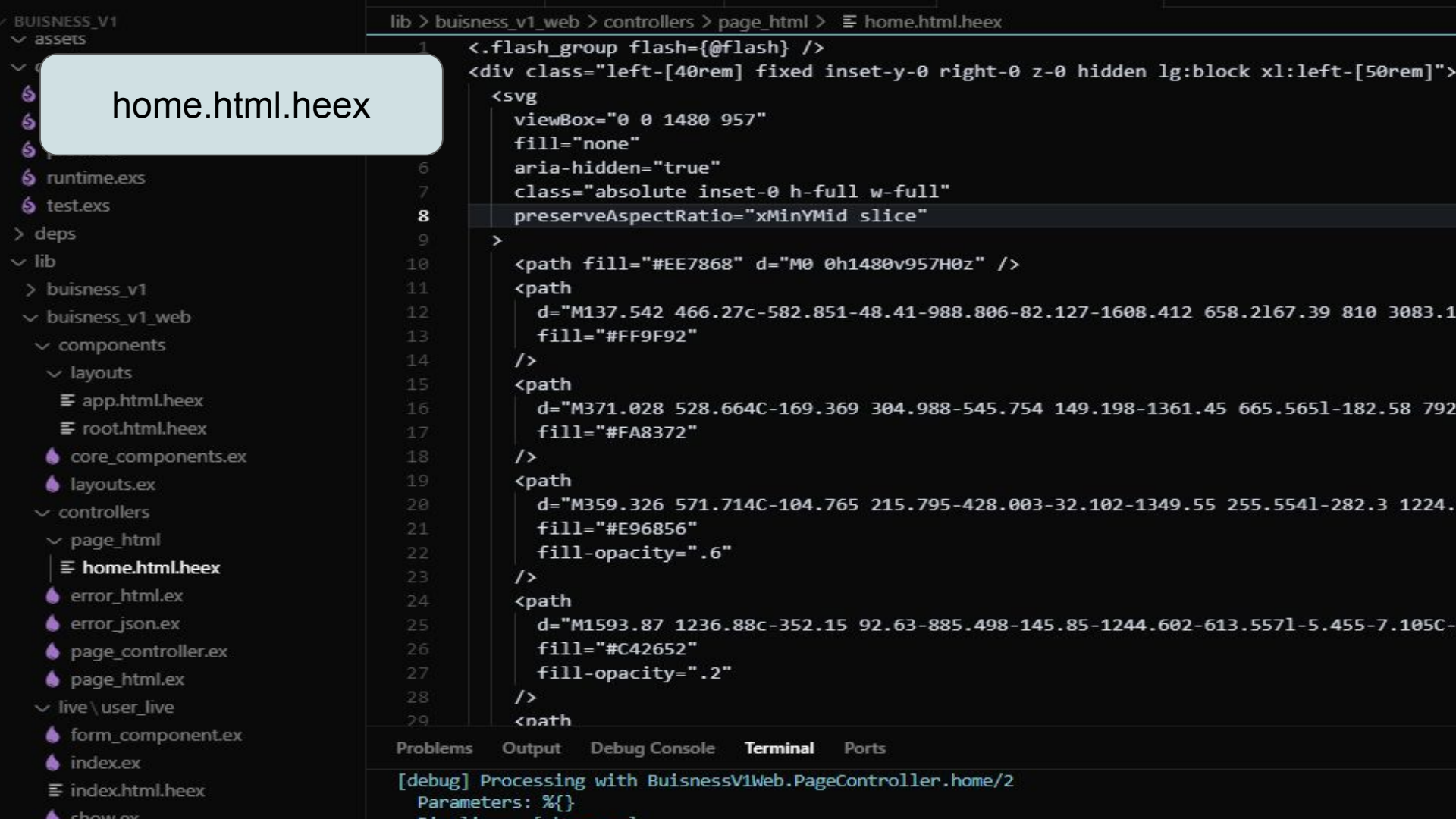
- Contains:

  - Static HTML (headings, text, images)

  - Embedded Elixir code (`<%= %>`) for dynamic content

  - Can include links, buttons, images, and LiveView component

  - Think of it as the main Homepage design


home.html.heex
BUISNESS_V1
assets
runtime.exs
test.exs
deps
lib
buisness_v1
buisness_v1_web
components
layouts
app.html.heex
root.html.heex
core_components.ex
layouts.ex
controllers
page_html
home.html.heex
error_html.ex
error_json.ex
page_controller.ex
page_html.ex
live \ user_live
form_component.ex
index.ex
index.html.heex
lib > buisness_v1_web > controllers > page_html > home.html.heex
<.flash_group flash={@flash} />

<svg
viewBox="0 0 1480 957"
fill="none"
aria-hidden="true"
class="absolute inset-0 h-full w-full"
preserveAspectRatio="xMinYMid slice"
>
<path fill="#EE7868" d="M0 0h1480v957H0z" />
<path
d="M137.542 466.27c-582.851-48.41-988.806-82.127-1608.412 658.2l67.39 810 3083.1
fill="#FF9F92"
/>
<path
d="M371.028 528.664C-169.369 304.988-545.754 149.198-1361.45 665.565l-182.58 792
fill="#FA8372"
/>
<path
d="M359.326 571.714C-104.765 215.795-428.003-32.102-1349.55 255.554l-282.3 1224.
fill="#E96856"
fill-opacity=".6"
/>
<path
d="M1593.87 1236.88c-352.15 92.63-885.498-145.85-1244.602-613.557l-5.455-7.105C-
fill="#C42652"
fill-opacity=".2"
/>
<path
Problems
Output
Debug Console
Terminal
Ports
[debug] Processing with BuisnessV1Web.PageController.home/2
Parameters: %{}

Phoenix Framework v1.7.21

Peace of mind from prototype to production.

Build rich, interactive web applications quickly, with less code and fewer moving parts. Join our growing community of developers using Phoenix to craft APIs, HTML5 apps and more, for fun or at scale.

Guides & Docs

Source Code

Changelog

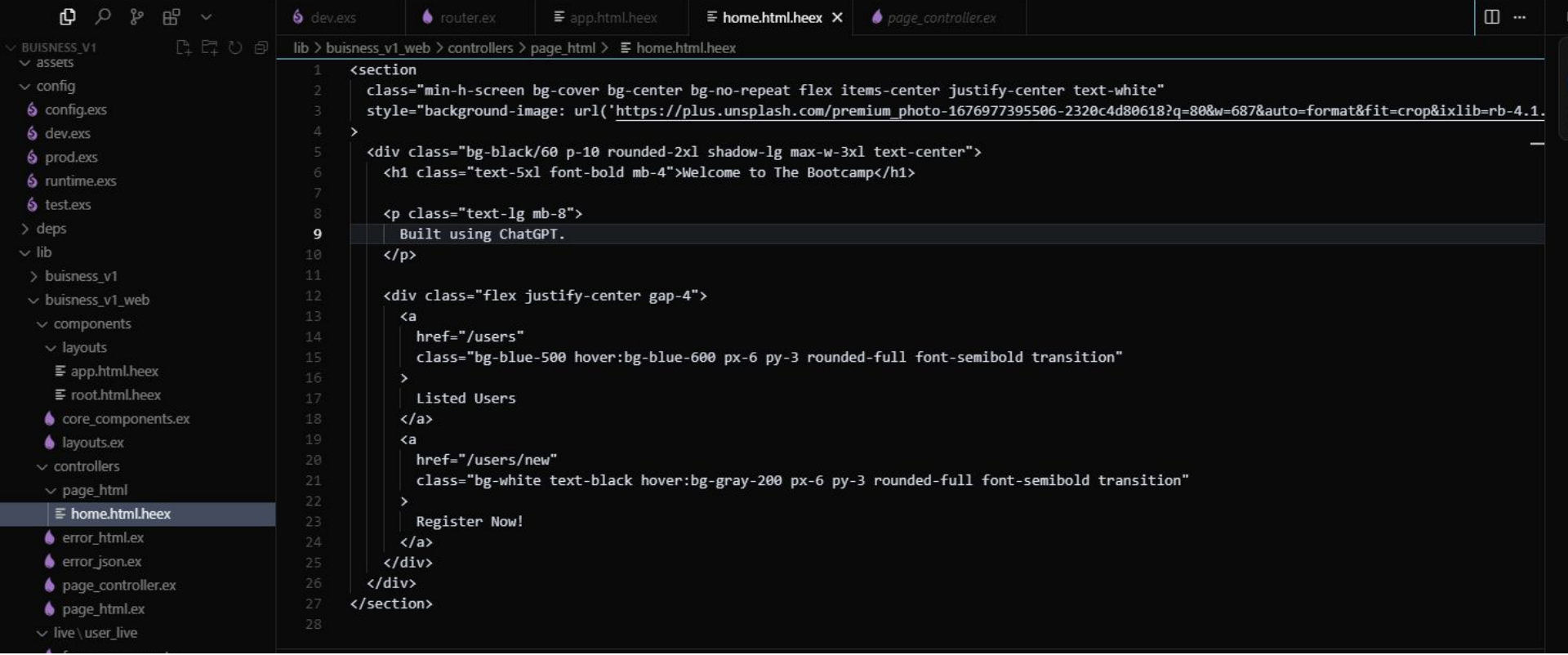
Follow on Twitter

Discuss on the Elixir Forum

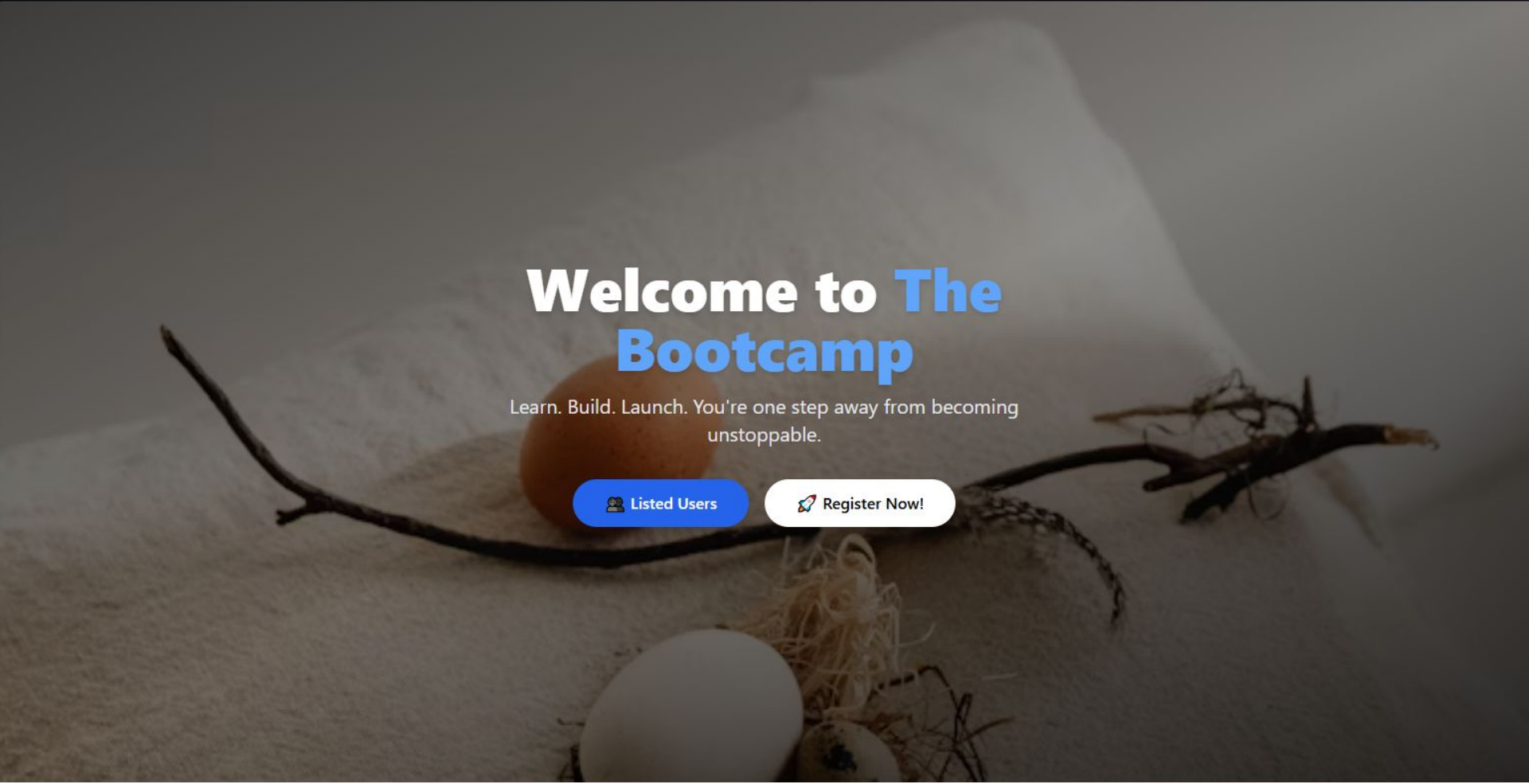
Chat on Libera IRC

Join our Discord server

Deploy your application



When you change something here..



The output will be on your Homepage.

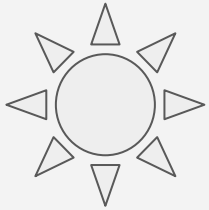
: Why Is It Important?

- First impression of your web app

- Central point for navigation

- Easy to customize with both HTML and Elixir

- Supports real-time updates when used with LiveView



This is why You must use a software like FIGMA

What is Figma?

Figma is a **web-based design tool** used for **UI/UX design**, **prototyping**, and **collaboration**. It allows designers, developers, and teams to create and test user interfaces and digital products — all in one place.