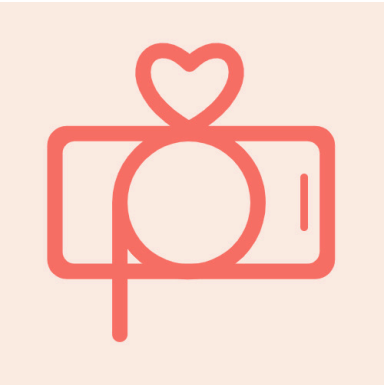
ENTWICKLUNG SERVERSEITIGER WEBANWENDUNGEN MIT HTMX UND SPRING BOOT

Thomas Schilling

Thomas Schilling (Schühly)

- Spring Boot + HTMX = ❤️
- Speaker: Spring IO, Devoxx, JavaZone, W-JAX, Java Forum Stuttgart, JUG Saxony, Paderborn, Frankfurt, Linz
- Founder PhotoQuest



htmx.tschuehly.de

Building server-side web applications with htmx

Lab 1: Server-side rendering with Spring Boot and JTE

Lab 2: Using Spring ViewComponent

Lab 3: Inline Editing

Lab 4: Using Spring Beans to Compose the UI

Lab 5: Lazy Loading

Lab 6: Full Text Search

Lab 7: Infinite Scroll

Lab 8: Exception Messages

Lab 9: Server-Sent Events

What today is about!

Who stays behind JSON API endpoints and doesn‘t touch the frontend?

Software Engineering

Software **Engineering**

“If an engineering approach to software development doesn’t help us to create better software faster, then it’s wrong and doesn’t qualify as “Engineering”. ”

Dave Farley

What is Modern Software Engineering?

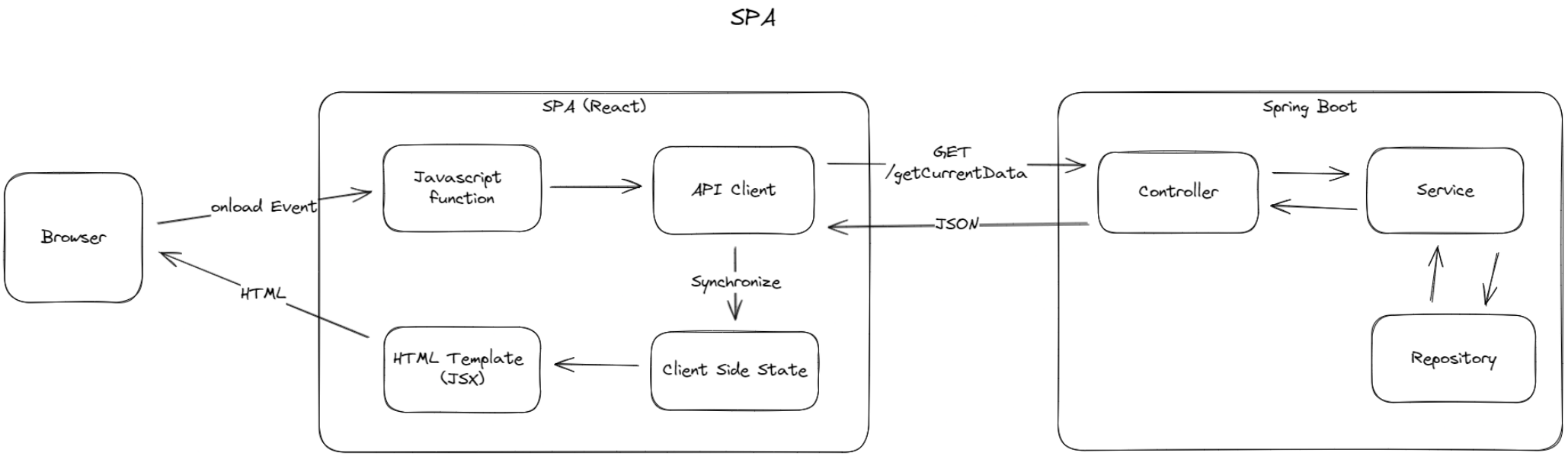
A deep dive into Cloudflare's September 12, 2025 dashboard and API outage

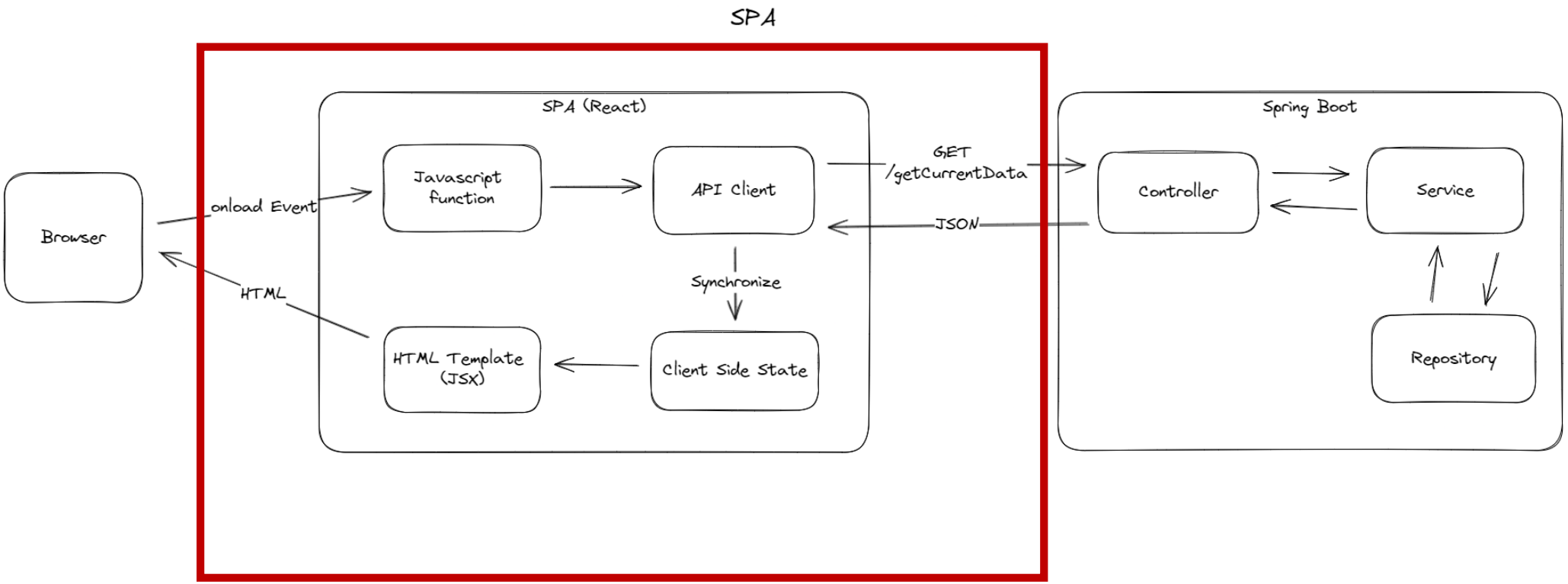
2025-09-13

The API calls were managed by a React useEffect hook, but we mistakenly included a problematic object in its dependency array.

https://blog.cloudflare.com/deep-dive-into-cloudflares-sept-12-dashboard-and-api-outage/

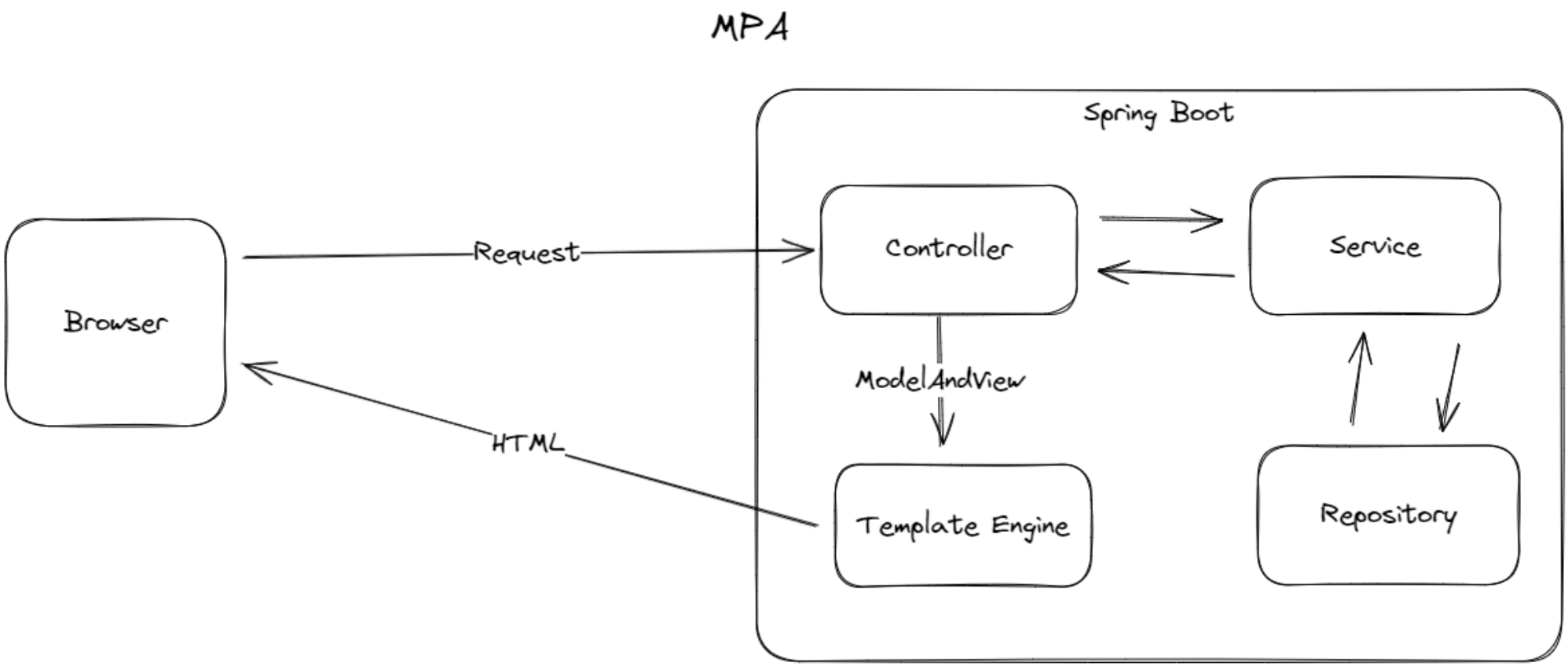
Single Page Applications



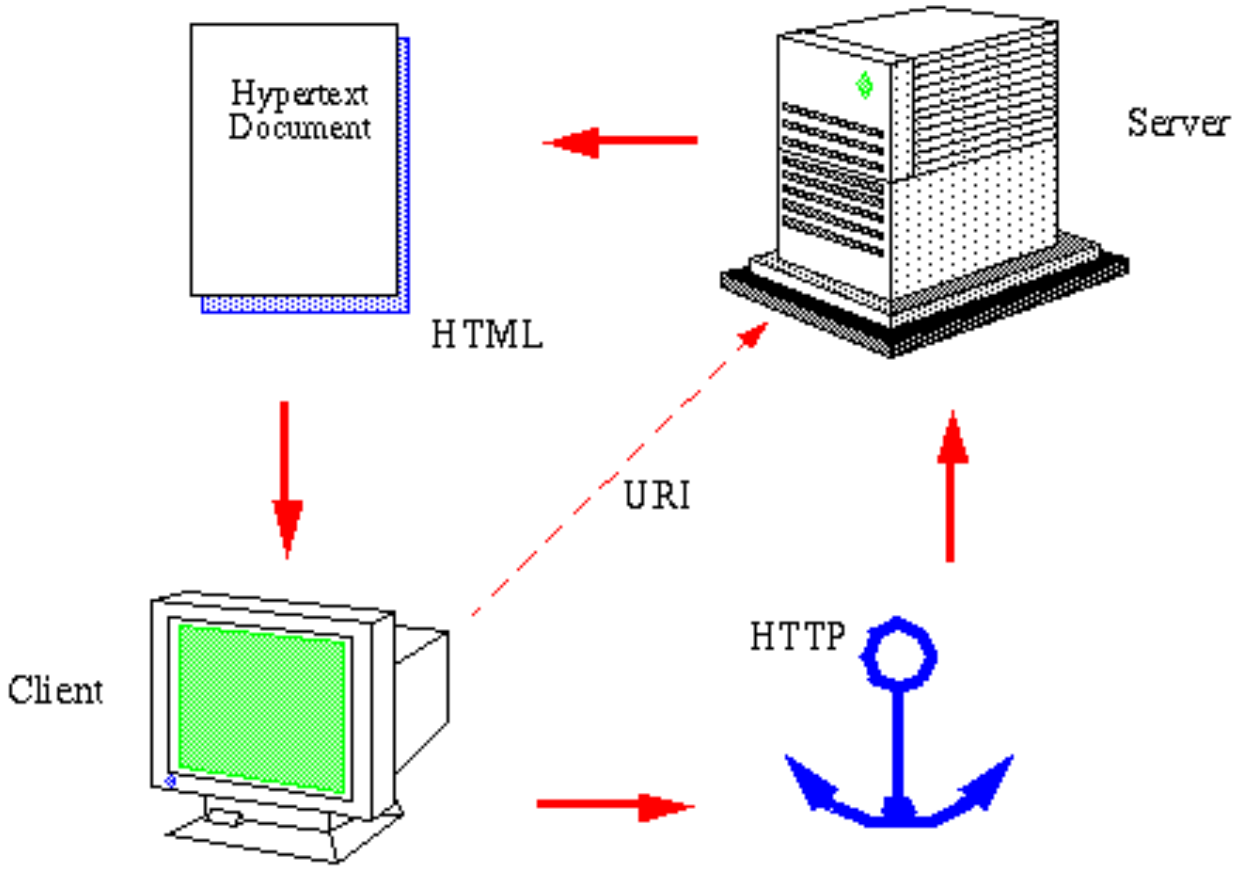

SPA
SPA (React)
Spring Boot
Browser
onload Event
HTML
Javascript function
API Client
GET /getCurrentData
JSON
Synchronize
HTML Template (JSX)
Client Side State
Controller
Service
Repository

- Single Page Applications are an architectural style, that incurs additional complexity.
- Tradeoffs review is skipped, if requirements need a framework like React

thoughtworks Technology Radar Vol. 27
www.thoughtworks.com/content/dam/thoughtworks/documents/radar/2022/10/tr_technology_radar_vol_27_en.pdf

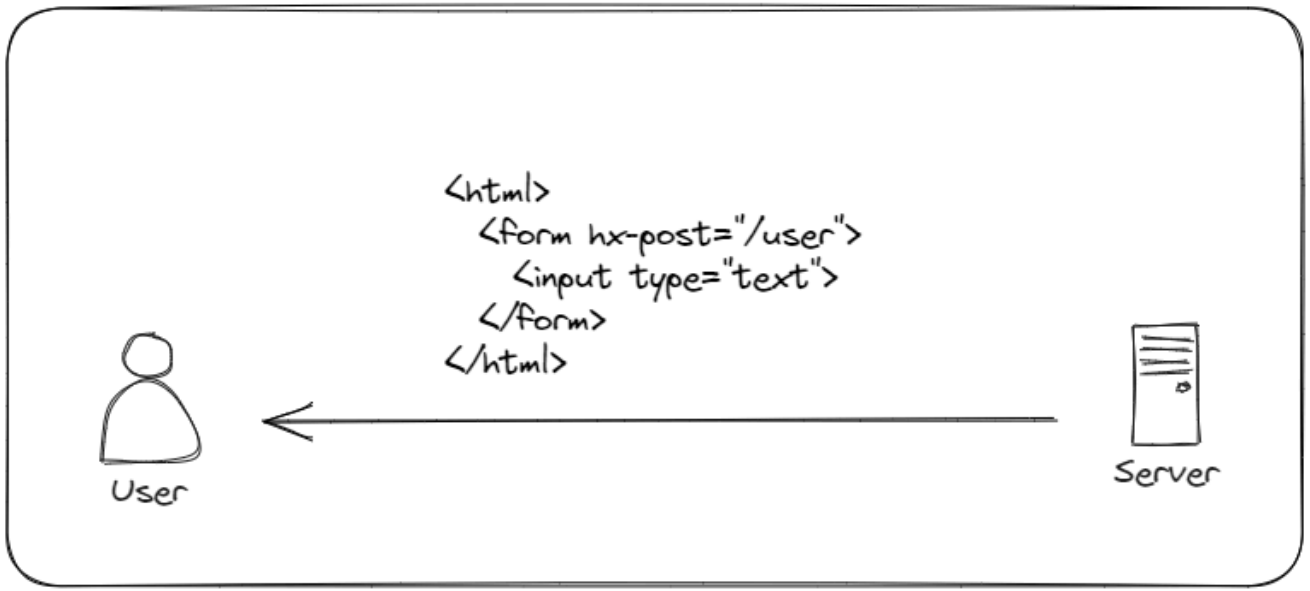

MPA
Spring Boot
Browser
Request
Controller
Service
ModelAndView
HTML
Template Engine
Repository

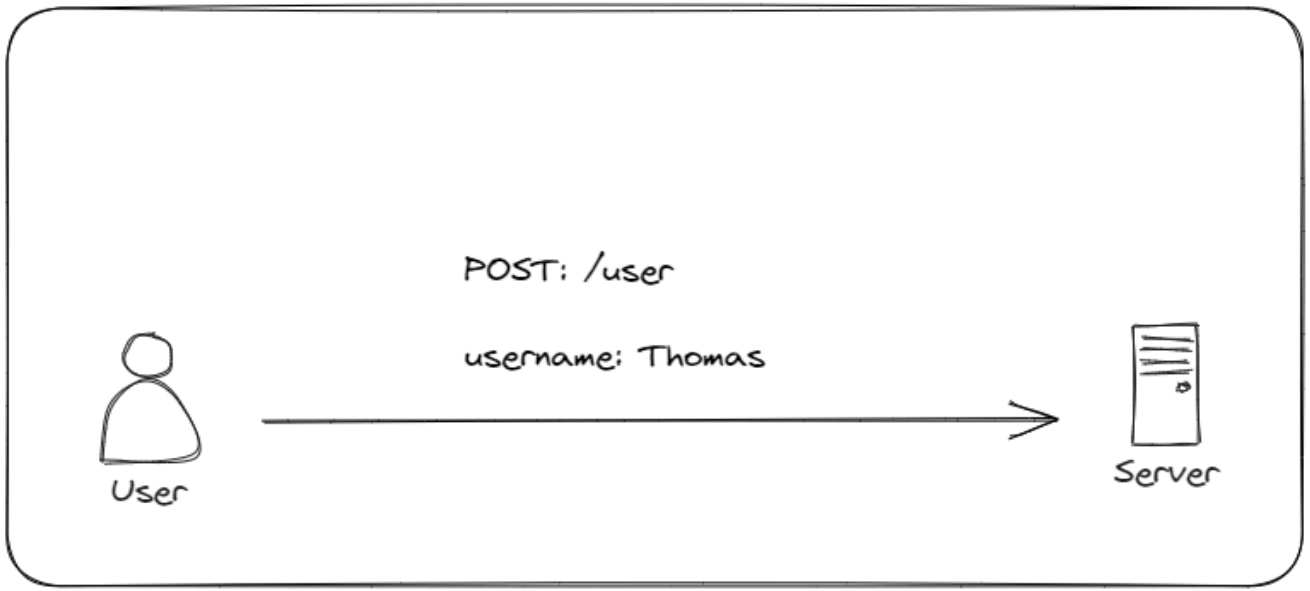

DB TO
JSON TO
JS TO HTML
DB
TO HTML
imgflip.com

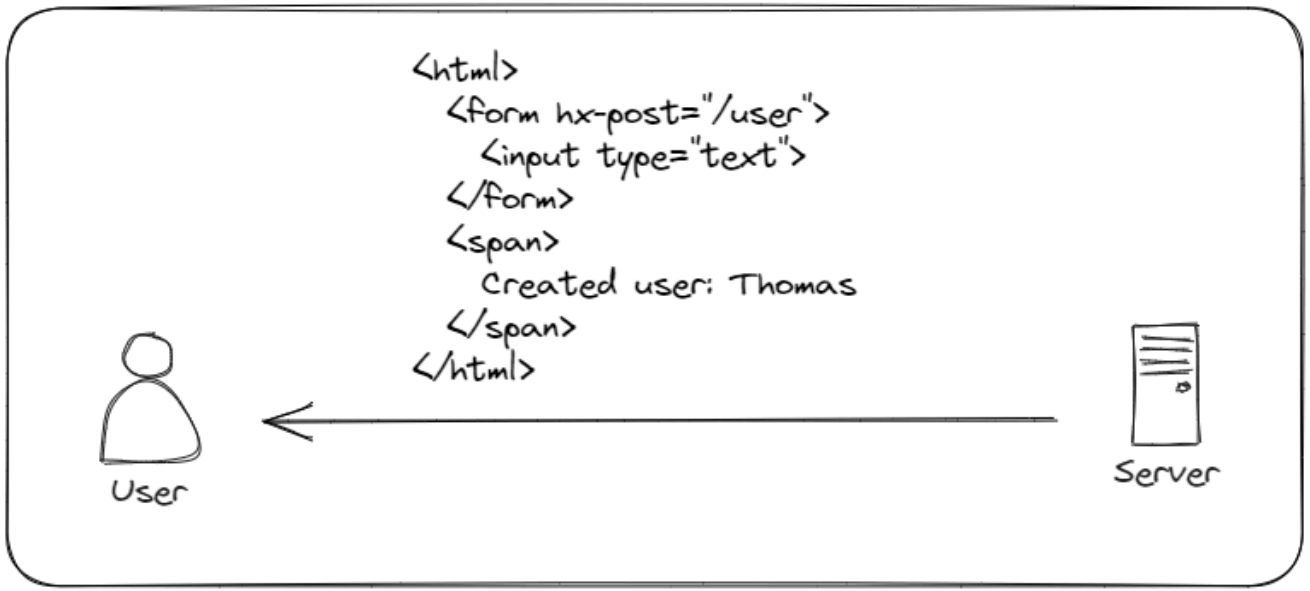

Hypertext
Document
HTML
Server
URI
Client
HTTP

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Website</title>
  </head>
  <body>
    <a href="http://example.com/">Go to example.com</a>
  </body>
</html>
```


GET /userform
User
Server


<html>
<form hx-post="/user">
<input type="text">
</form>
</html>
User
Server


POST: /user
username: Thomas
User
Server


<html>
<form hx-post="/user">
<input type="text">
</form>

Created user: Thomas

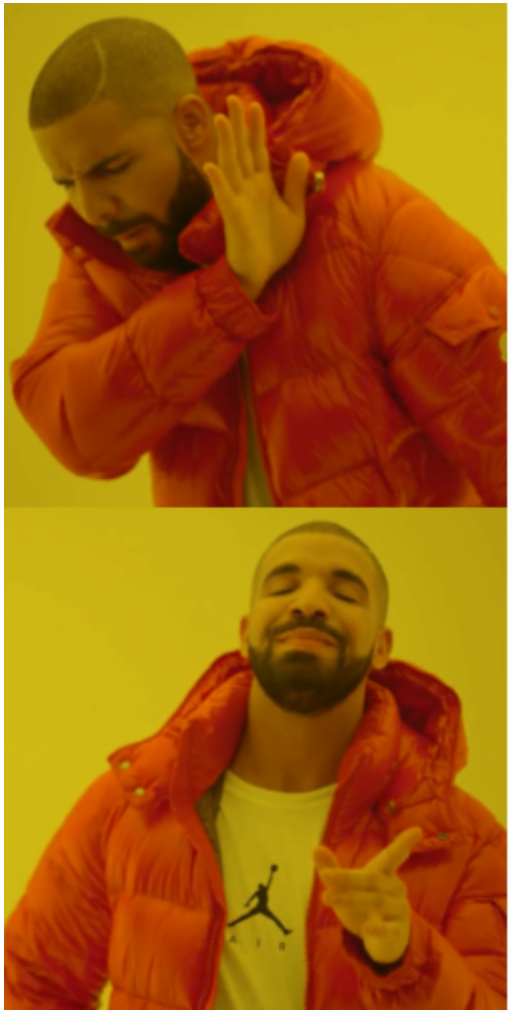
</html>
User
Server

https://roy.gbiv.com/untangled/2008/rest-apis-must-be-hypertext-driven

Benefits

- Hypermedia allows your application API to be much more aggressively refactored and optimized
- Hypermedia is often significantly less complex than an SPA approach would be for many problems

https://htmx.org/essays/when-to-use-hypermedia/



Creating your own Client

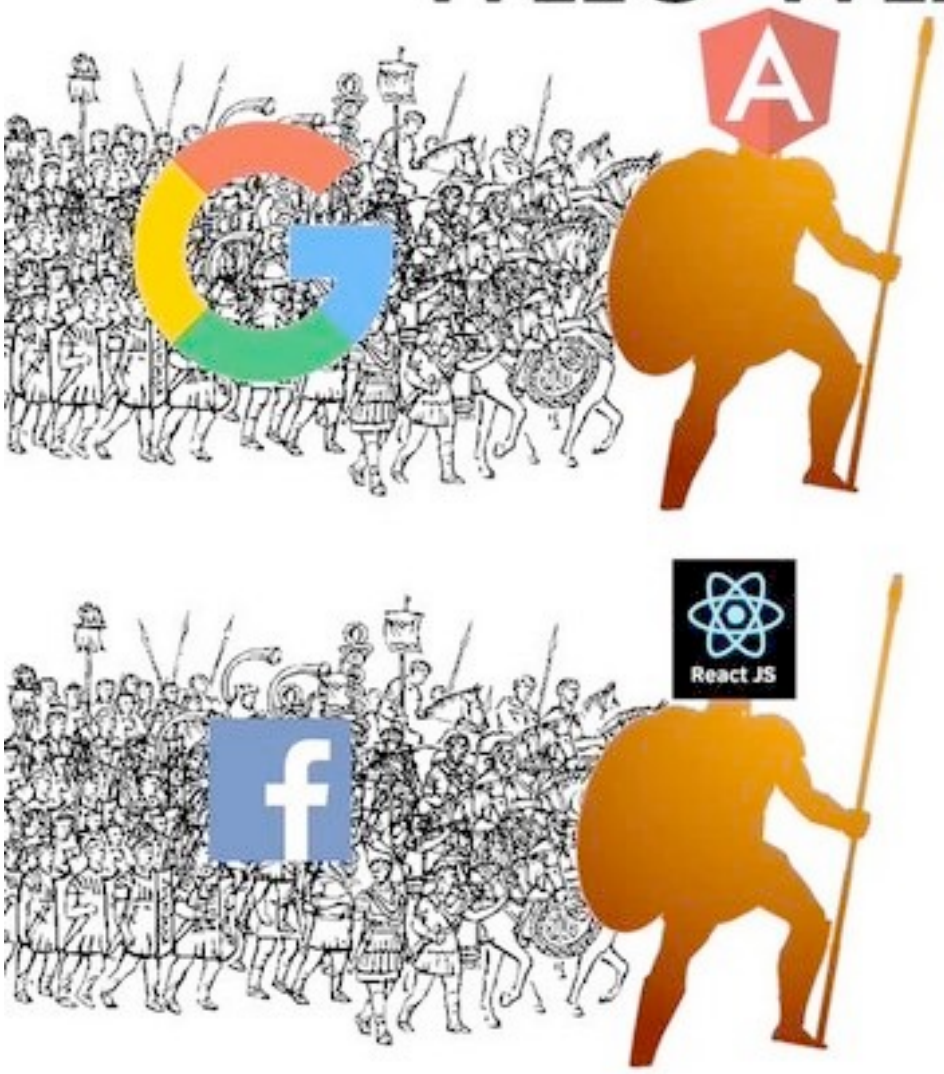
Just use the Browser as Client

8.2.4. Example Form Submission: Questionnaire Form

Consider the following document:

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<title>Sample of HTML Form Submission</title>
<H1>Sample Questionnaire</H1>
<P>Please fill out this questionnaire:
<FORM METHOD="POST" ACTION="http://www.w3.org/sample">
<P>Your name: <INPUT NAME="name" size="48">
<P>Male <INPUT NAME="gender" TYPE=RADIO VALUE="male">
<P>Female <INPUT NAME="gender" TYPE=RADIO VALUE="female">
<P>Number in family: <INPUT NAME="family" TYPE=text>
<P>Cities in which you maintain a residence:
<UL>
<LI>Kent <INPUT NAME="city" TYPE=checkbox VALUE="kent">
<LI>Miami <INPUT NAME="city" TYPE=checkbox VALUE="miami">
<LI>Other <TEXTAREA NAME="other" cols=48 rows=4></textarea>
</UL>
Nickname: <INPUT NAME="nickname" SIZE="42">
<P>Thank you for responding to this questionnaire.
<P><INPUT TYPE=SUBMIT> <INPUT TYPE=RESET>
</FORM>
```

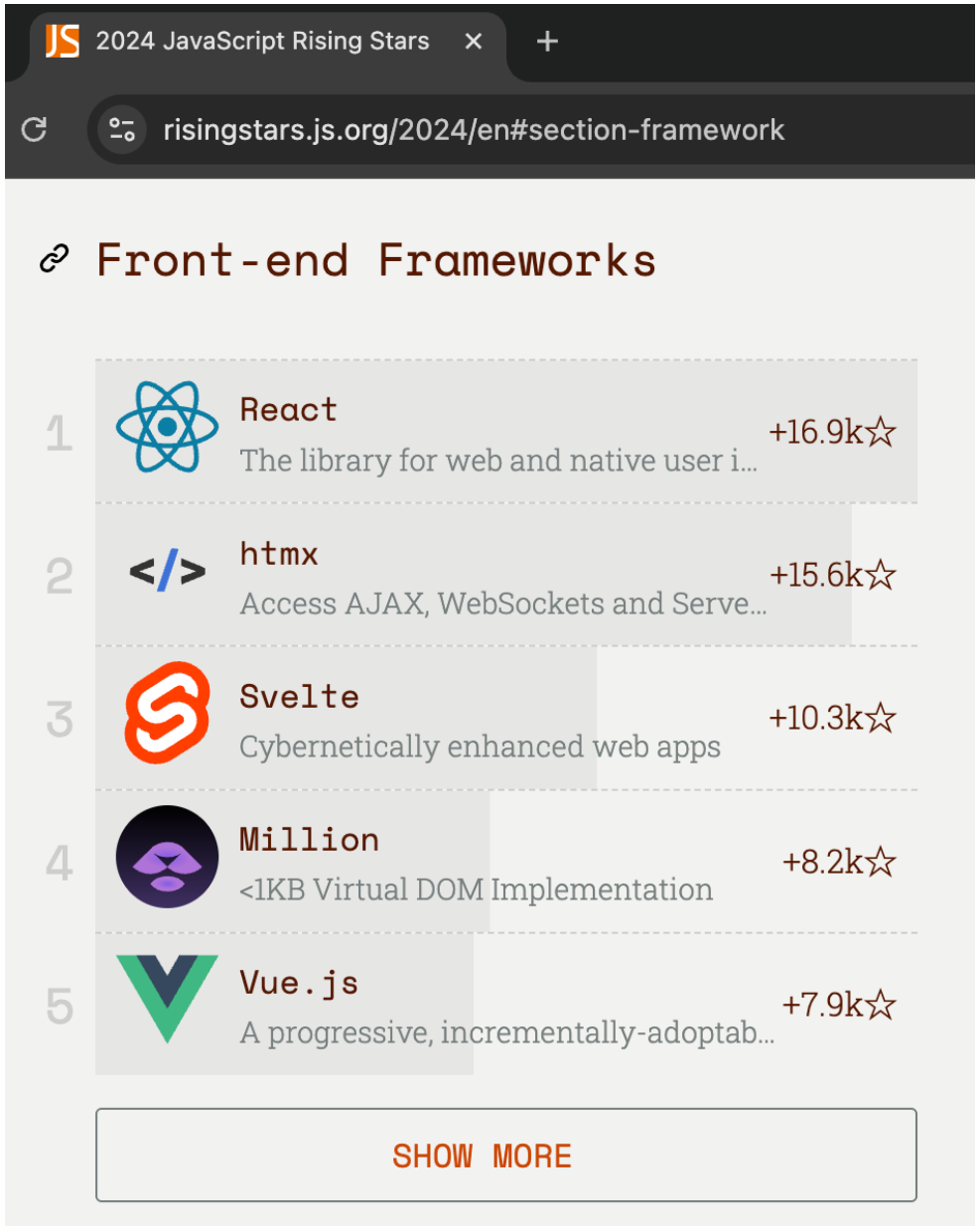
WHO WILL WIN?

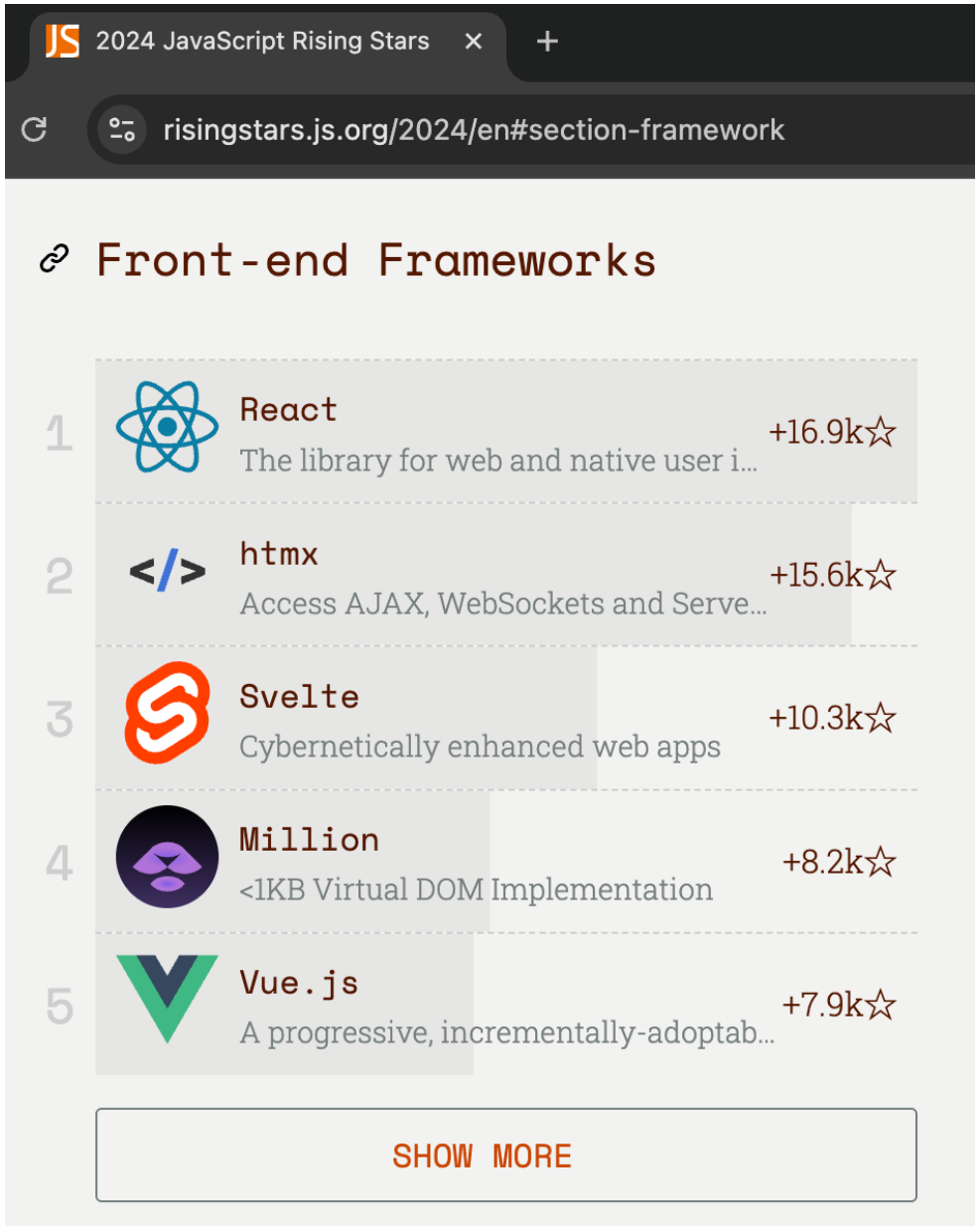


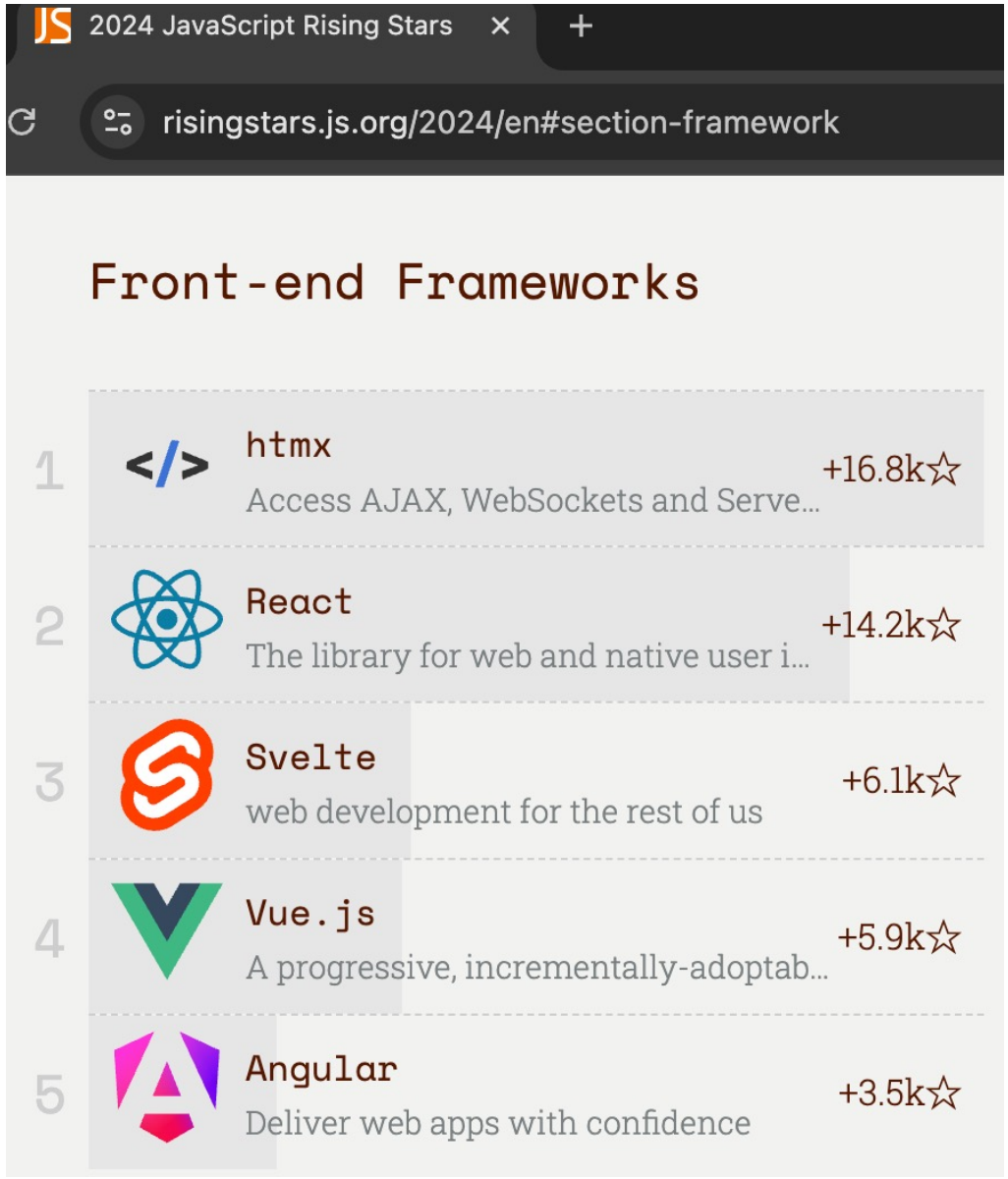
VS.

a lunatic in montana and his internet friends


htmx
high power tools for HTML


2024 JavaScript Rising Stars
risingstars.js.org/2024/en#section-framework
Front-end Frameworks
1
React
The library for web and native user i...
+16.9k☆
2
htmx
Access AJAX, WebSockets and Serve...
+15.6k☆
3
Svelte
Cybernetically enhanced web apps
+10.3k☆
4
Million
<1KB Virtual DOM Implementation
+8.2k☆
5
Vue.js
A progressive, incrementally-adoptab...
+7.9k☆
SHOW MORE


2024 JavaScript Rising Stars
risingstars.js.org/2024/en#section-framework
Front-end Frameworks
1
React
The library for web and native user i...
+16.9k☆
2
htmx
Access AJAX, WebSockets and Serve...
+15.6k☆
3
Svelte
Cybernetically enhanced web apps
+10.3k☆
4
Million
<1KB Virtual DOM Implementation
+8.2k☆
5
Vue.js
A progressive, incrementally-adoptab...
+7.9k☆
SHOW MORE


2024 JavaScript Rising Stars
risingstars.js.org/2024/en#section-framework
Front-end Frameworks
1
htmx
Access AJAX, WebSockets and Serve...
+16.8k☆
2
React
The library for web and native user i...
+14.2k☆
3
Svelte
web development for the rest of us
+6.1k☆
4
Vue.js
A progressive, incrementally-adoptab...
+5.9k☆
5
Angular
Deliver web apps with confidence
+3.5k☆

```
<head>
    <script src="/htmx_1.9.11.js"></script>
</head>
```

```
<a href="/blog">Blog</a>
```


<button hx-post="/clicked"
hx-trigger="click"
hx-target="#parent-div"
hx-swap="outerHTML" >
Click Me!
</button>


LOOK AT ME
BACKEND DEV
HTMX
YOU'RE FULLSTACK NOW
imgflip.com
MC$

haiku

javascript fatigue:
longing for a hypertext
already in hand

Our Speakers

Talk

David Guillot

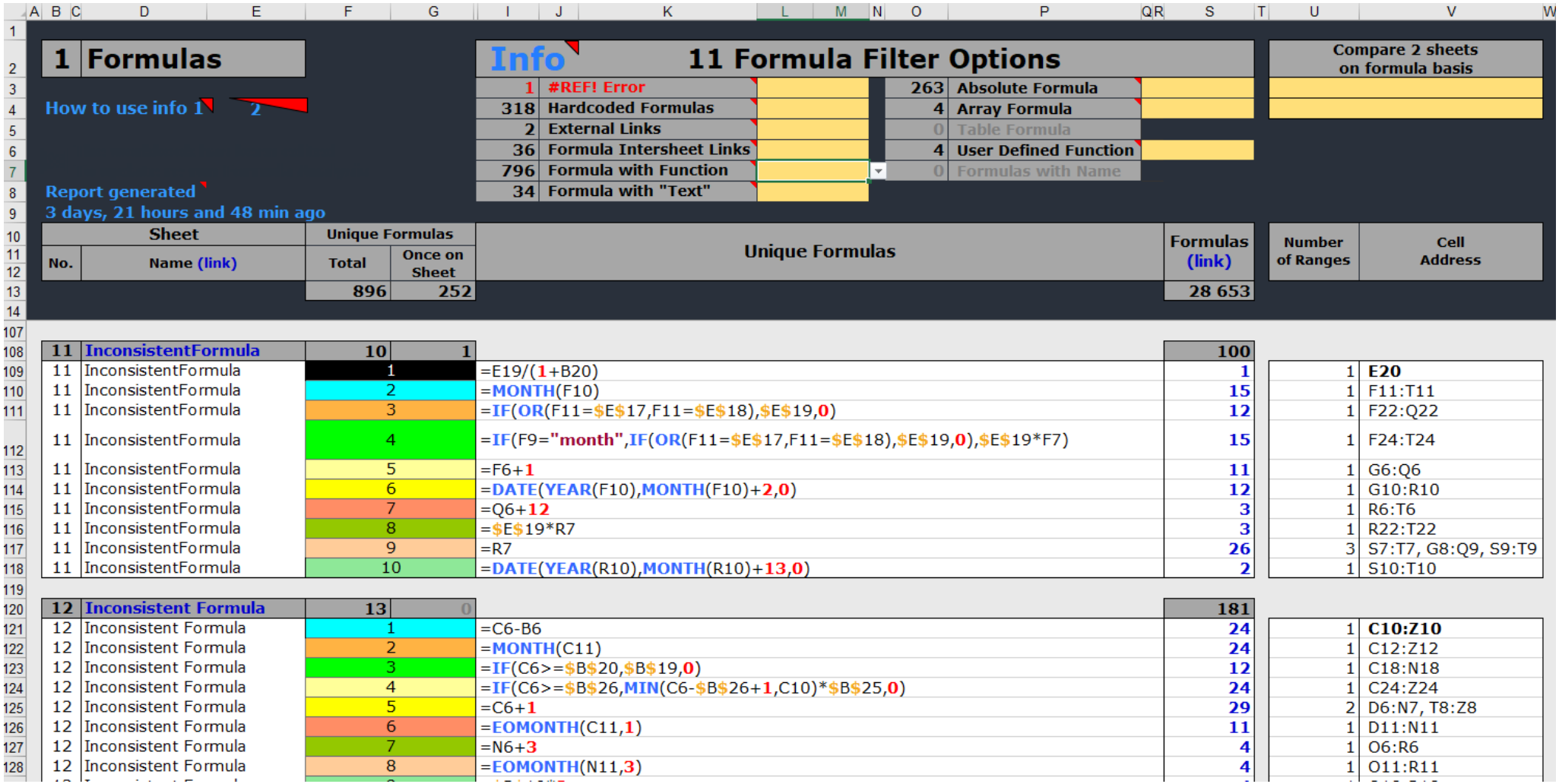
From React to htmx on a real-world SaaS product: we did it, and it's awesome!

2022 djangocon.eu porto

SEPTEMBER 21ST-25TH

htmx.org/examples/active-search

	HDA (htmx)	SPA (React)
CRUD / Formulare	Ideal	Overkill
Content-lastige UIs	Ideal	Unnötig komplex
Abgegrenzte UI-Updates	Ideal	Auch gut
Echtzeit-Kollaboration	Begrenzt	Besser geeignet
Offline-Fähigkeit	Nicht möglich	Gut möglich
Komplexe UI-Abhängigkeiten	Schwierig	Besser geeignet
SEO / First Render	Nativ	Erfordert SSR-Setup



1 Formulas

How to use info 1 2

Report generated
3 days, 21 hours and 48 min ago

Info

11 Formula Filter Options

1	#REF! Error		263	Absolute Formula	
318	Hardcoded Formulas		4	Array Formula	
2	External Links		0	Table Formula	
36	Formula Intersheet Links		4	User Defined Function	
796	Formula with Function		0	Formulas with Name	
34	Formula with "Text"				

Compare 2 sheets on formula basis

Sheet		Unique Formulas		Unique Formulas	Formulas (link)	Number of Ranges	Cell Address
No.	Name (link)	Total	Once on Sheet				
		896	252		28 653		
11	InconsistentFormula	10	1		100		
11	InconsistentFormula	1		=E19/(1+B20)	1	1	**E20**
11	InconsistentFormula	2		=MONTH(F10)	15	1	F11:T11
11	InconsistentFormula	3		=IF(OR(F11=$E$17,F11=$E$18),$E$19,0)	12	1	F22:Q22
11	InconsistentFormula	4		=IF(F9="month",IF(OR(F11=$E$17,F11=$E$18),$E$19,0),$E$19*F7)	15	1	F24:T24
11	InconsistentFormula	5		=F6+1	11	1	G6:Q6
11	InconsistentFormula	6		=DATE(YEAR(F10),MONTH(F10)+2,0)	12	1	G10:R10
11	InconsistentFormula	7		=Q6+12	3	1	R6:T6
11	InconsistentFormula	8		=$E$19*R7	3	1	R22:T22
11	InconsistentFormula	9		=R7	26	3	S7:T7, G8:Q9, S9:T9
11	InconsistentFormula	10		=DATE(YEAR(R10),MONTH(R10)+13,0)	2	1	S10:T10
12	Inconsistent Formula	13	0		181		
12	Inconsistent Formula	1		=C6-B6	24	1	**C10:Z10**
12	Inconsistent Formula	2		=MONTH(C11)	24	1	C12:Z12
12	Inconsistent Formula	3		=IF(C6>=$B$20,$B$19,0)	12	1	C18:N18
12	Inconsistent Formula	4		=IF(C6>=$B$26,MIN(C6-$B$26+1,C10)*$B$25,0)	24	1	C24:Z24
12	Inconsistent Formula	5		=C6+1	29	2	D6:N7, T8:Z8
12	Inconsistent Formula	6		=EOMONTH(C11,1)	11	1	D11:N11
12	Inconsistent Formula	7		=N6+3	4	1	O6:R6
12	Inconsistent Formula	8		=EOMONTH(N11,3)	4	1	O11:R11

When Should You Use Hypermedia?

Carson Gross

October 23, 2022

> The trade-off, though, is that a uniform interface degrades efficiency, since information is transferred in a standardized form rather than one which is specific to an application's needs. The REST interface is designed to be efficient for large-grain hypermedia data transfer, optimizing for the common case of the Web, but resulting in an interface that is not optimal for other forms of architectural interaction.

~90% aller Business-Webanwendungen sind CRUD mit UI drumherum

Nutzerverwaltung

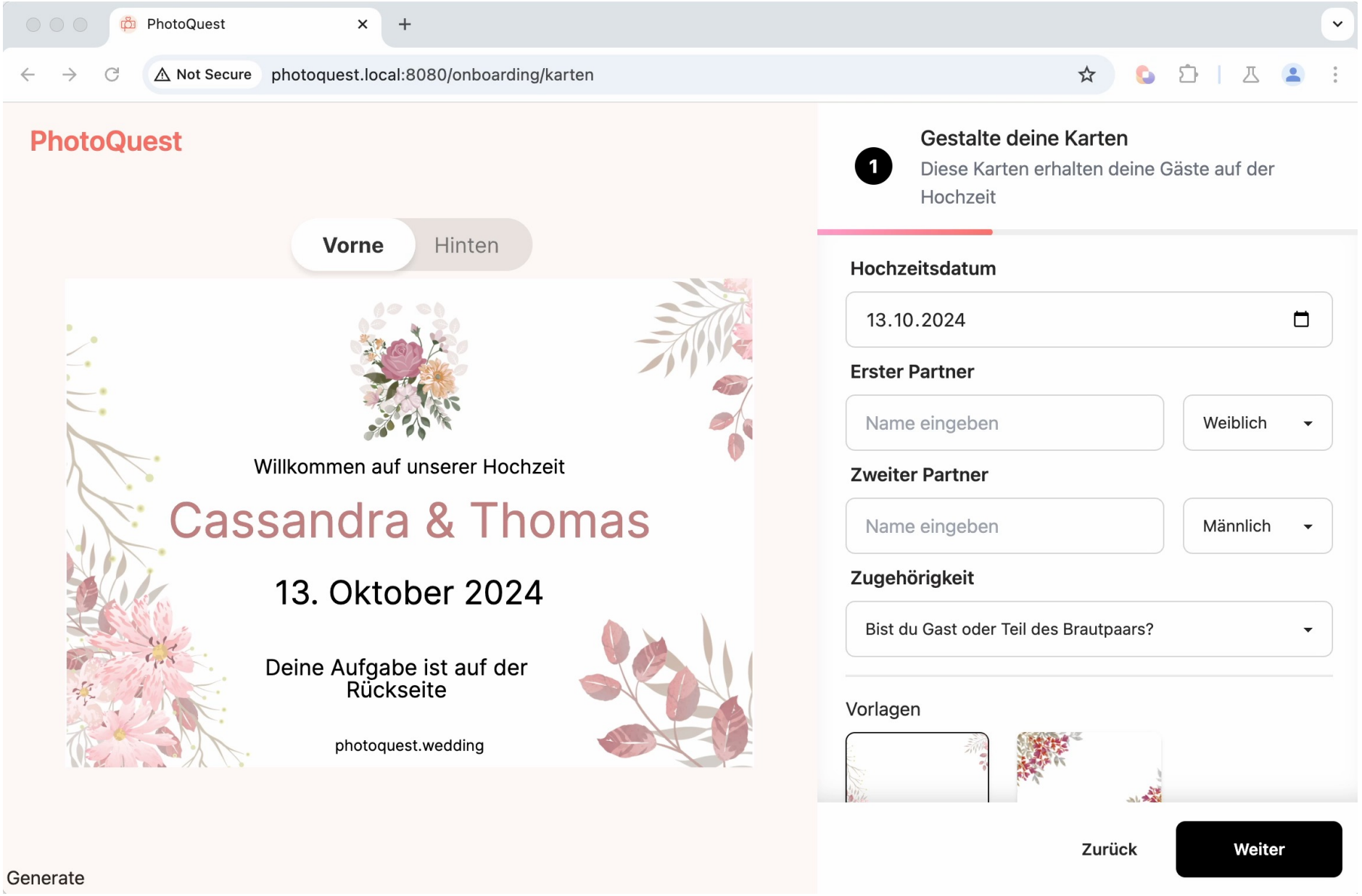
Bestellsystem

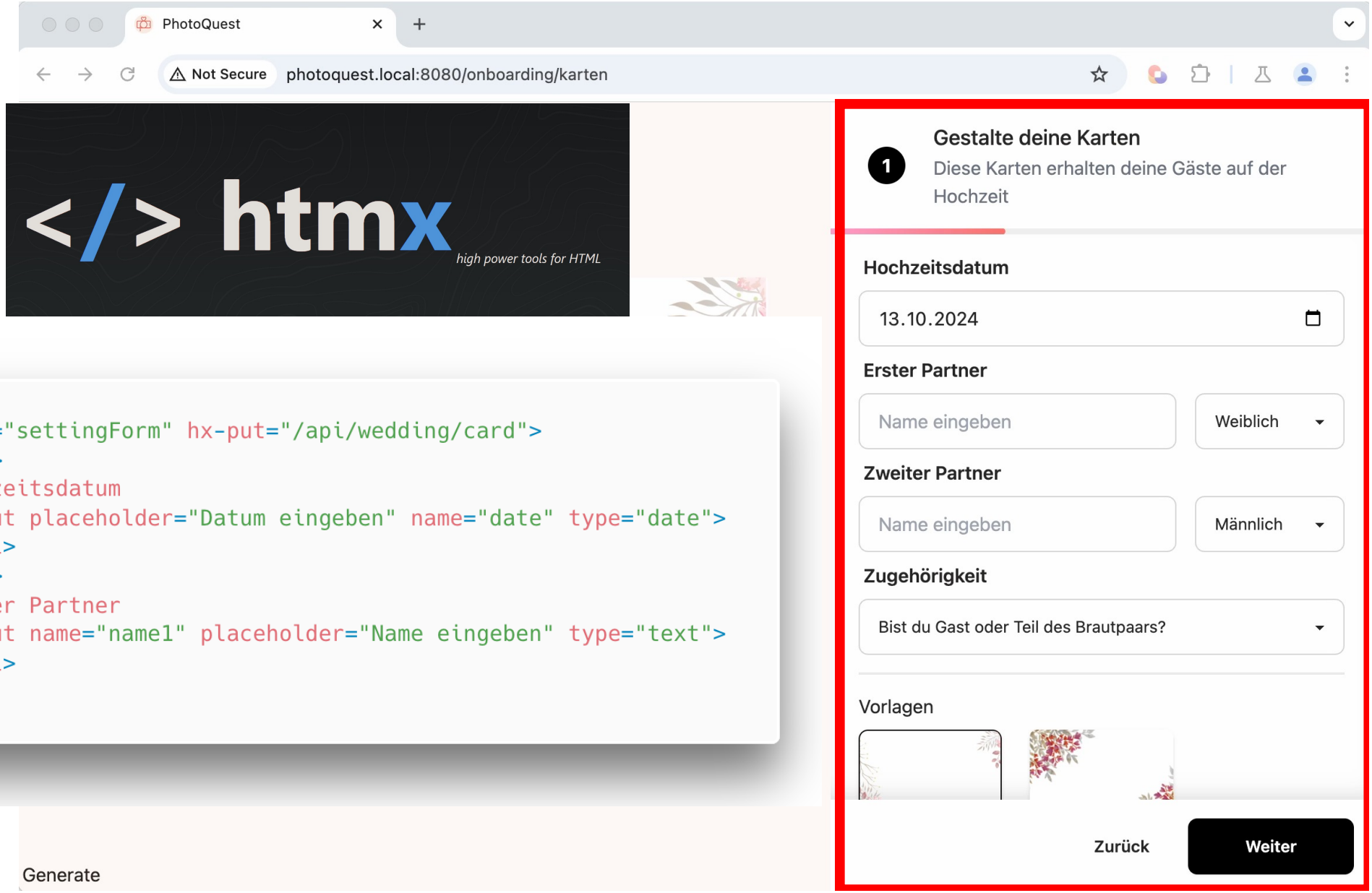
Ticket-System (CRUD)

Internes Admin-Tool (CRUD)

Transitional Applications

- Adopting Hypermedia is not an either/or decision
- Use the right tool for the right use case


PhotoQuest
Not Secure
photoquest.local:8080/onboarding/karten
PhotoQuest
Vorne
Hinten
Willkommen auf unserer Hochzeit
Cassandra & Thomas
13. Oktober 2024
Deine Aufgabe ist auf der Rückseite
photoquest.wedding
Generate
1
Gestalte deine Karten
Diese Karten erhalten deine Gäste auf der Hochzeit
Hochzeitsdatum
13.10.2024
Erster Partner
Name eingeben
Weiblich
Zweiter Partner
Name eingeben
Männlich
Zugehörigkeit
Bist du Gast oder Teil des Brautpaars?
Vorlagen
Zurück
Weiter


PhotoQuest
Not Secure
photoquest.local:8080/onboarding/karten
</> htmx
high power tools for HTML
<form id="settingForm" hx-put="/api/wedding/card">
<label>
Hochzeitsdatum
<input placeholder="Datum eingeben" name="date" type="date">
</label>
<label>
Erster Partner
<input name="name1" placeholder="Name eingeben" type="text">
</label>
</form>
Generate
1
Gestalte deine Karten
Diese Karten erhalten deine Gäste auf der Hochzeit
Hochzeitsdatum
13.10.2024
Erster Partner
Name eingeben
Weiblich
Zweiter Partner
Name eingeben
Männlich
Zugehörigkeit
Bist du Gast oder Teil des Brautpaars?
Vorlagen
Zurück
Weiter

PhotoQuest

Not Secure photoquest.local:8080/onboarding/karten

PhotoQuest

Vorne Hinten

Generate

1 Gestalte deine Karten

Diese Karten erhalten deine Gäste auf der

pdfme: Free and Open source PDF generation library!

A powerful PDF generation library fully written in TypeScript, featuring a React-based UI template editor for seamless PDF creation.

Zurück Weiter

PhotoQuest

Not Secure photoquest.local:8080/onboarding/karten

PhotoQuest

Vorne Hinten

Generate

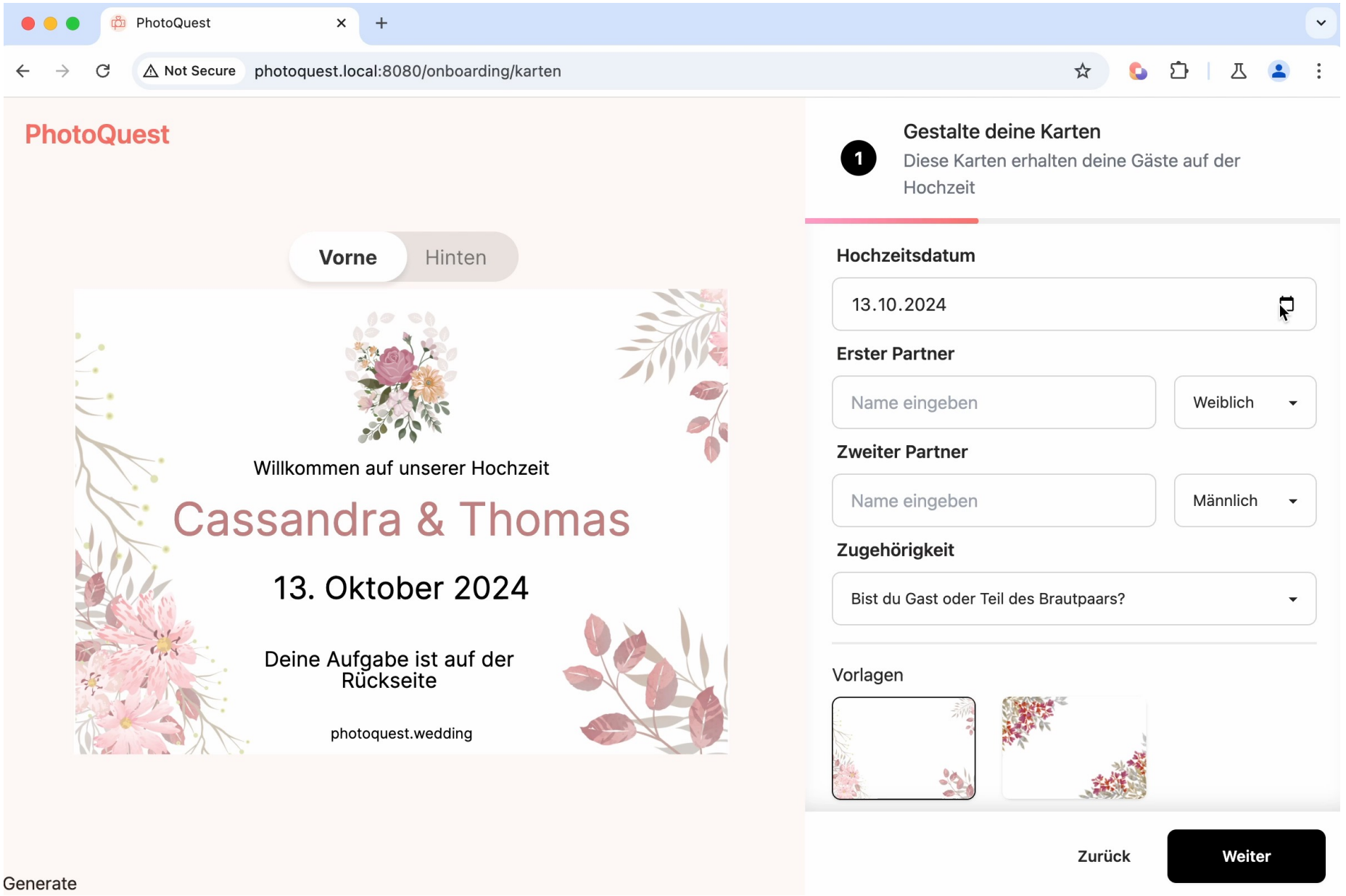
1 Gestalte deine Karten

Diese Karten erhalten deine Gäste auf der

pdfme: Free and Open source PDF generation library!

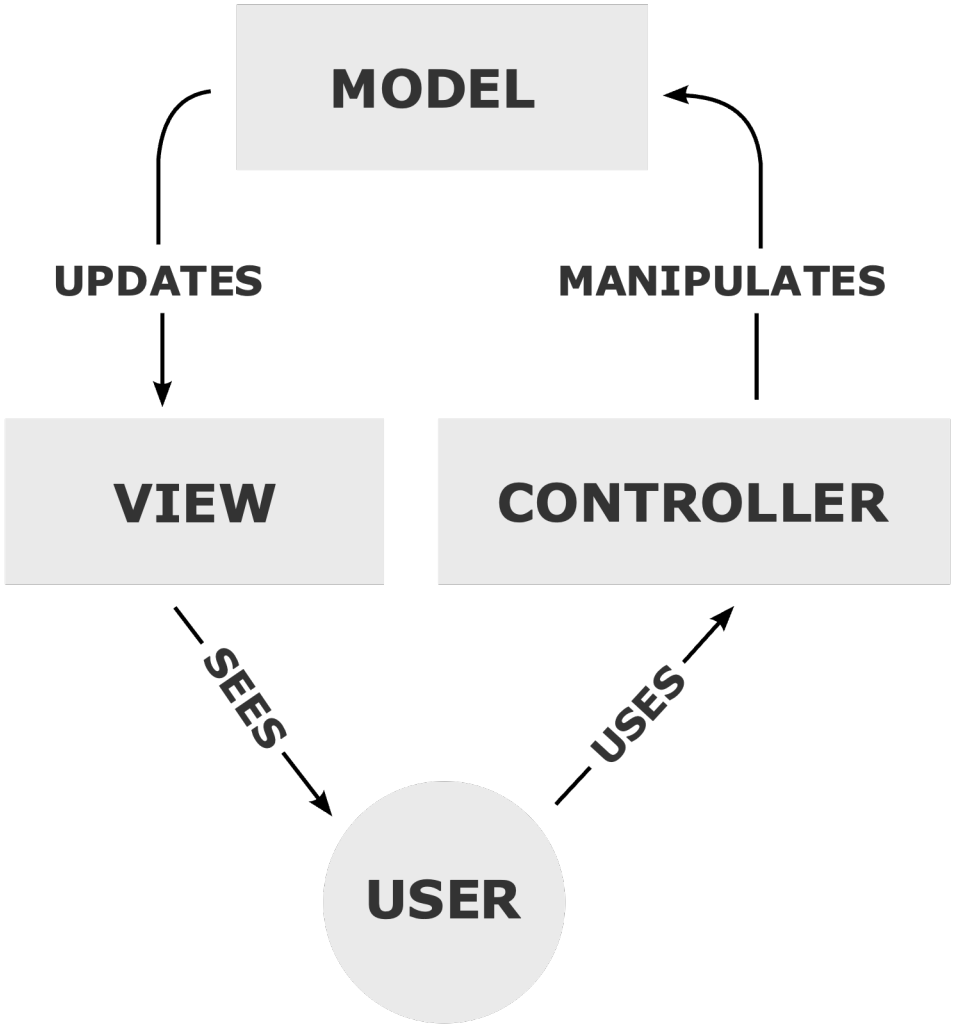
A powerful PDF generation library fully written in TypeScript, featuring a React-based UI template editor for seamless PDF creation.

Zurück Weiter

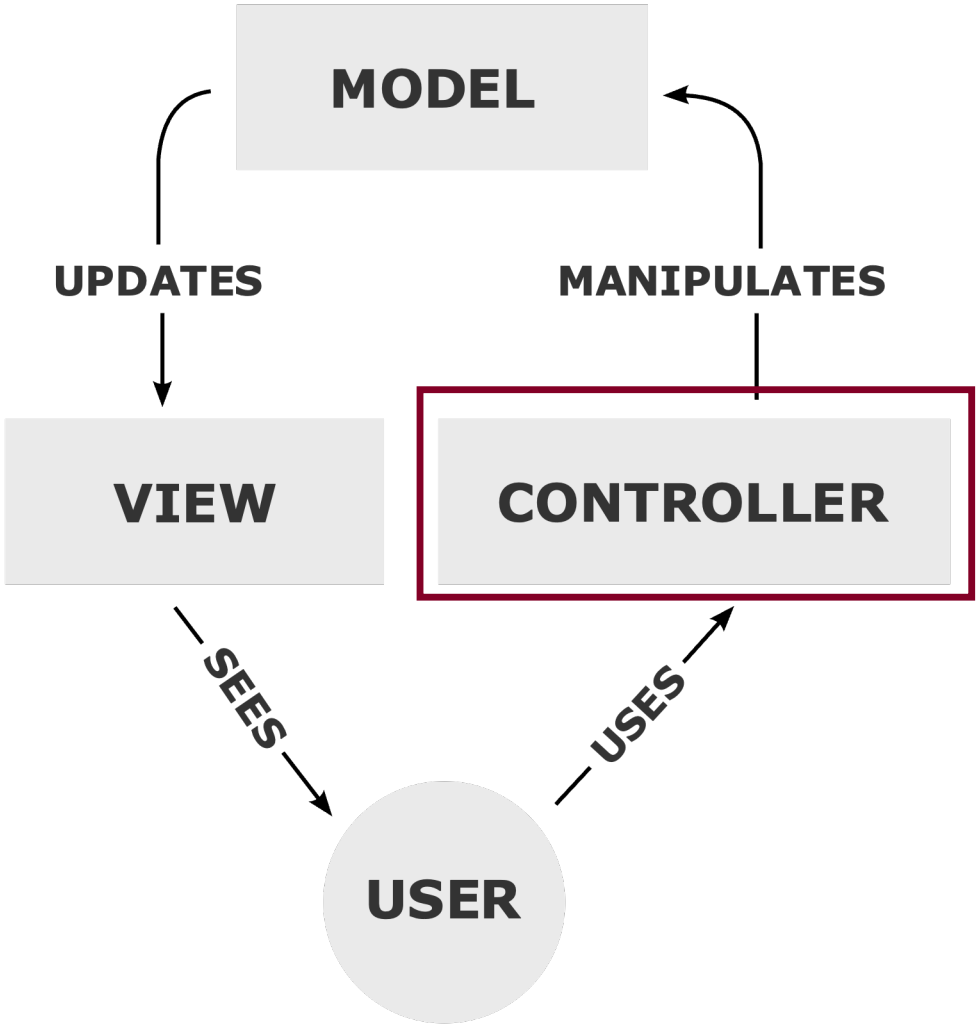

PhotoQuest
Not Secure
photoquest.local:8080/onboarding/karten
PhotoQuest
Vorne
Hinten
Willkommen auf unserer Hochzeit
Cassandra & Thomas
13. Oktober 2024
Deine Aufgabe ist auf der Rückseite
photoquest.wedding
1
Gestalte deine Karten
Diese Karten erhalten deine Gäste auf der Hochzeit
Hochzeitsdatum
13.10.2024
Erster Partner
Name eingeben
Weiblich
Zweiter Partner
Name eingeben
Männlich
Zugehörigkeit
Bist du Gast oder Teil des Brautpaars?
Vorlagen
Zurück
Weiter
Generate


spring

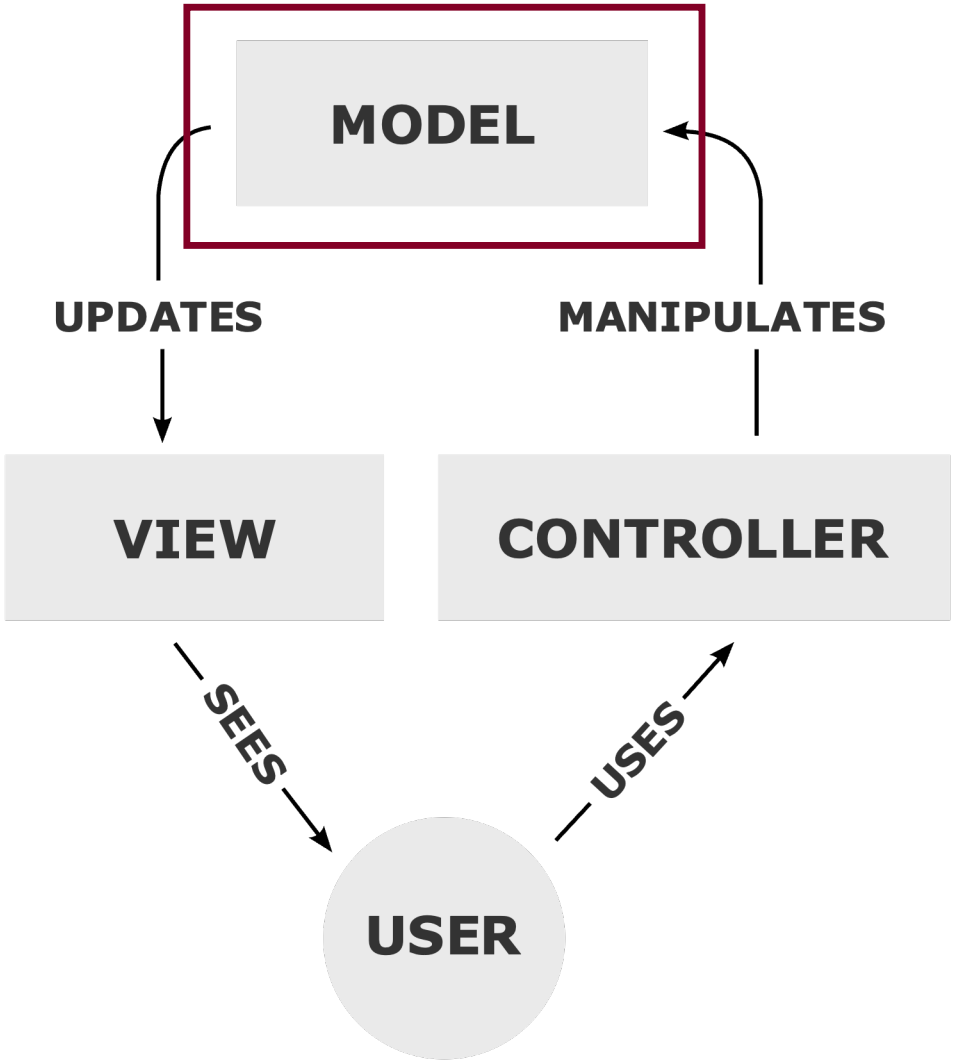

```
1 @Controller
2 class GreetingController {
3
4     @GetMapping("/greeting")
5     fun home(model: Model): String{
6
7         model.addAttribute("name", "SpringIO")
8
9         return "greeting"
10    }
11 }
```



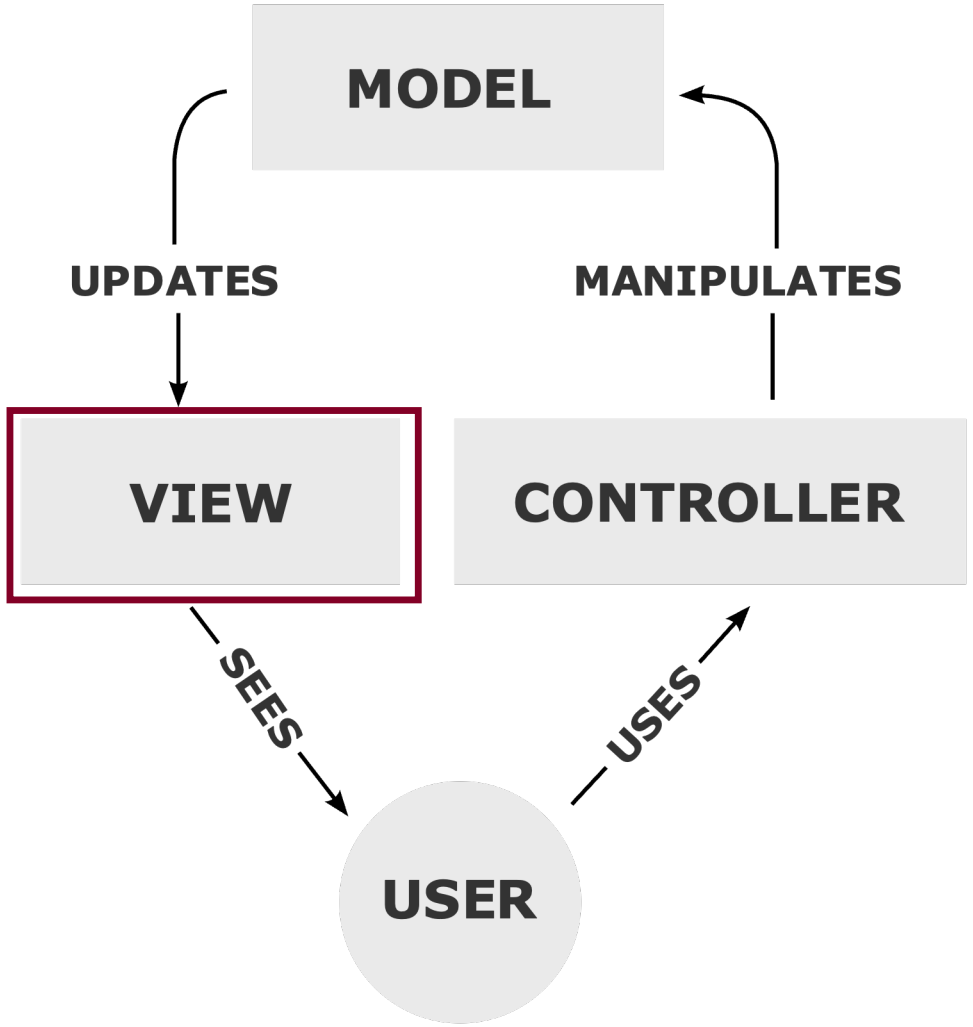

```kotlin
@Controller
class GreetingController {

    @GetMapping("/greeting")
    fun home(model: Model): String{

        model.addAttribute("name", "SpringIO")

        return "greeting"
    }
}
```



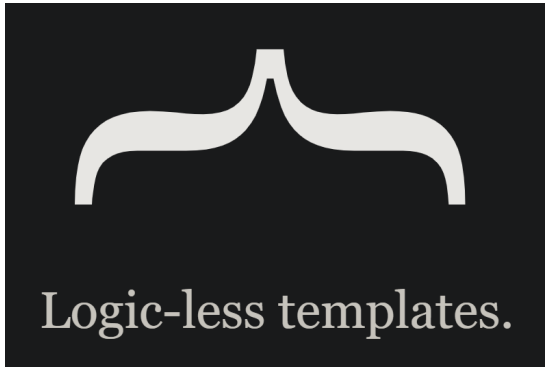

```
1 @Controller
2 class GreetingController {
3
4     @GetMapping("/greeting")
5     fun home(model: Model): String{
6
7         model.addAttribute("name", "SpringIO")
8
9         return "greeting"
10    }
11 }
```




```
1 @Controller
2 class GreetingController {
3
4     @GetMapping("/greeting")
5     fun home(model: Model): String{
6
7         model.addAttribute("name", "SpringIO")
8
9         return "greeting"
10    }
11 }
```




jstachio


Logic-less templates.


Thymeleaf


<#FREEMARKER>


jte

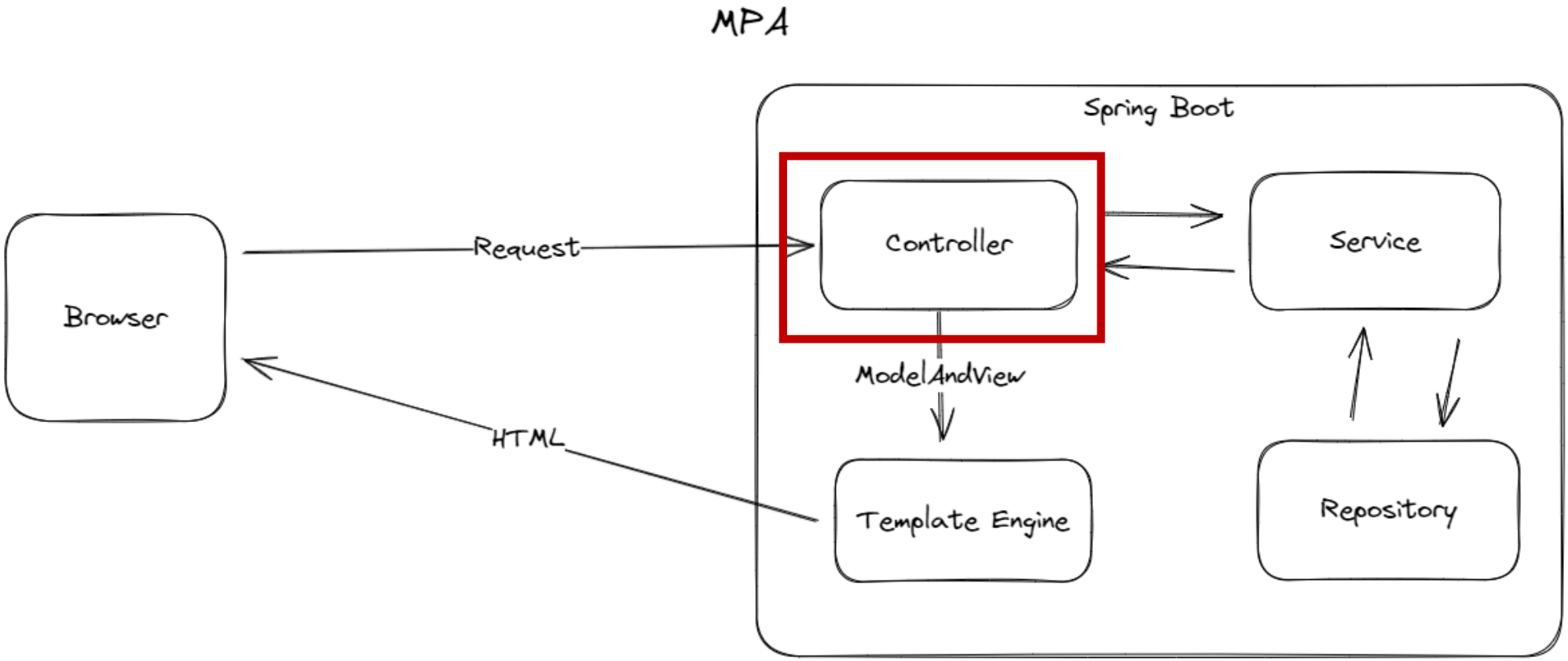

@import org.example.Page
@param Page page
<head>
<meta name="description"
content="${page.getDescription()}">
<title>${page.getTitle()}</title>
</head>


@import org.example.Page
@param Page page
<head>
<meta name="description"
content="${page.getDescription()}">
<title>${page.getTitle()}</title>
</head>

```
1 @import de.tschuehly.jteviewcomponentdemo.core.Person
2 @import java.util.List
3 @param List<Person> personList
4
5 <table>
6     <thead>
7         <tr>
8             <td>Id</td>
9             <td>Name</td>
10            <td>LastName</td>
11        </tr>
12    </thead>
13    <tbody>
14     @for(Person person : personList)
15         <tr>
16             <th>${person.getId()}</th>
17             <th>${person.getName()}</th>
18             <th>${person.getLastname()}</th>
19         </tr>
20     @endfor
21    </tbody>
22 </table>
```

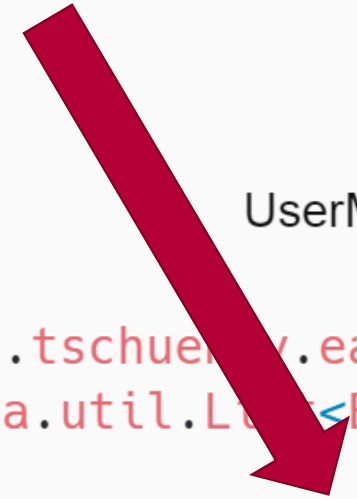
```
@import org.example.Page
@param Page page

<head>
    @if(page.getDescription() != null)
        <meta name="description" content="${page.getDescription()}">
    @endif
    <title>${page.getTitle()}</title>
</head>
<body>
    <h1>${page.getTitle()}</h1>
    <p>Welcome to my example page!</p>
</body>
```


spring


MPA
Spring Boot
Controller
Service
Browser
Request
ModelAndView
HTML
Template Engine
Repository

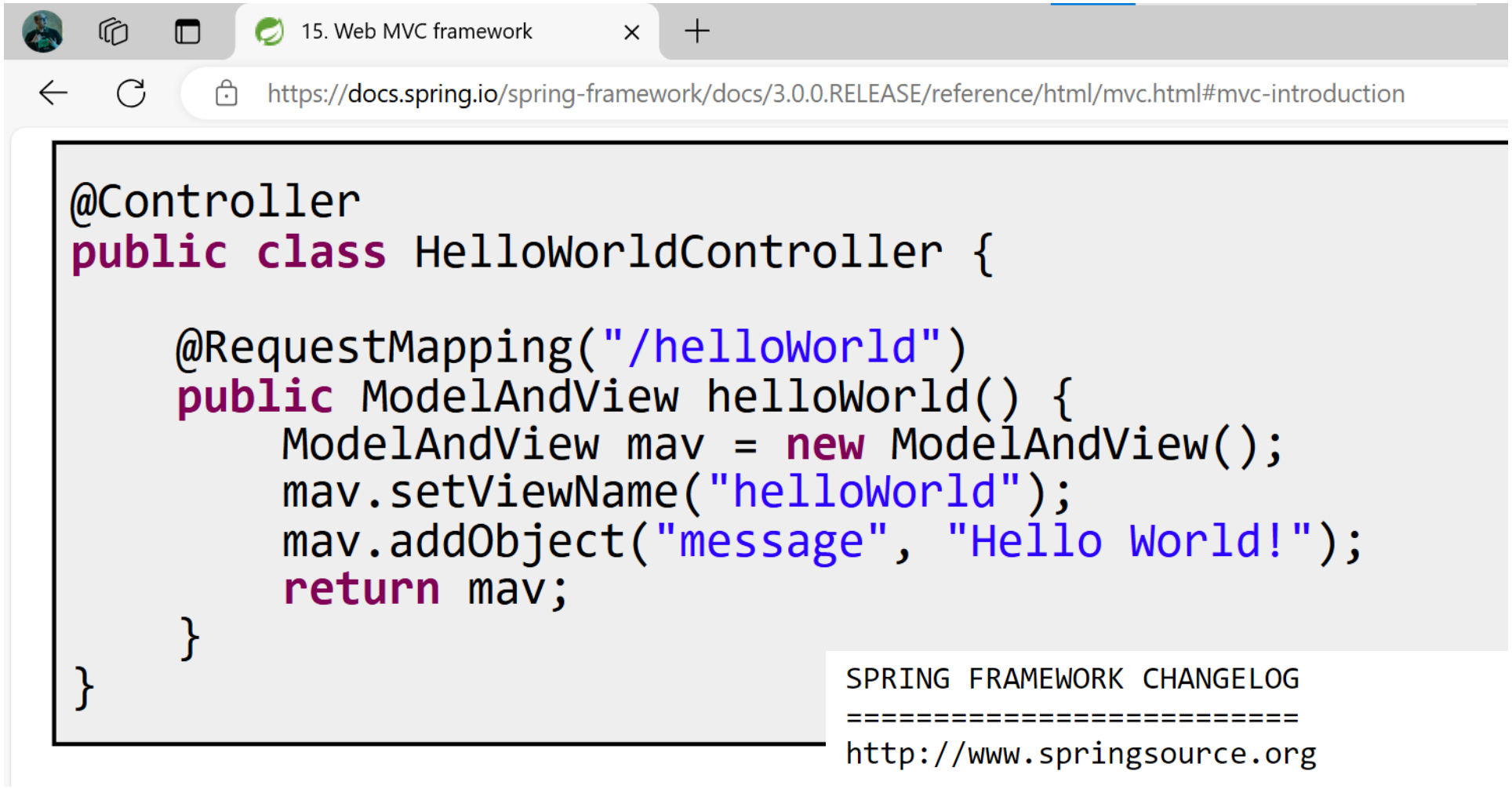
UserController.java

```java
@Controller
public class UserController {
  @GetMapping("/")
  public String index(Model model) {
    model.addAttribute("easyUserList", userService.findAll());
    return "UserManagement";
  }
}
```

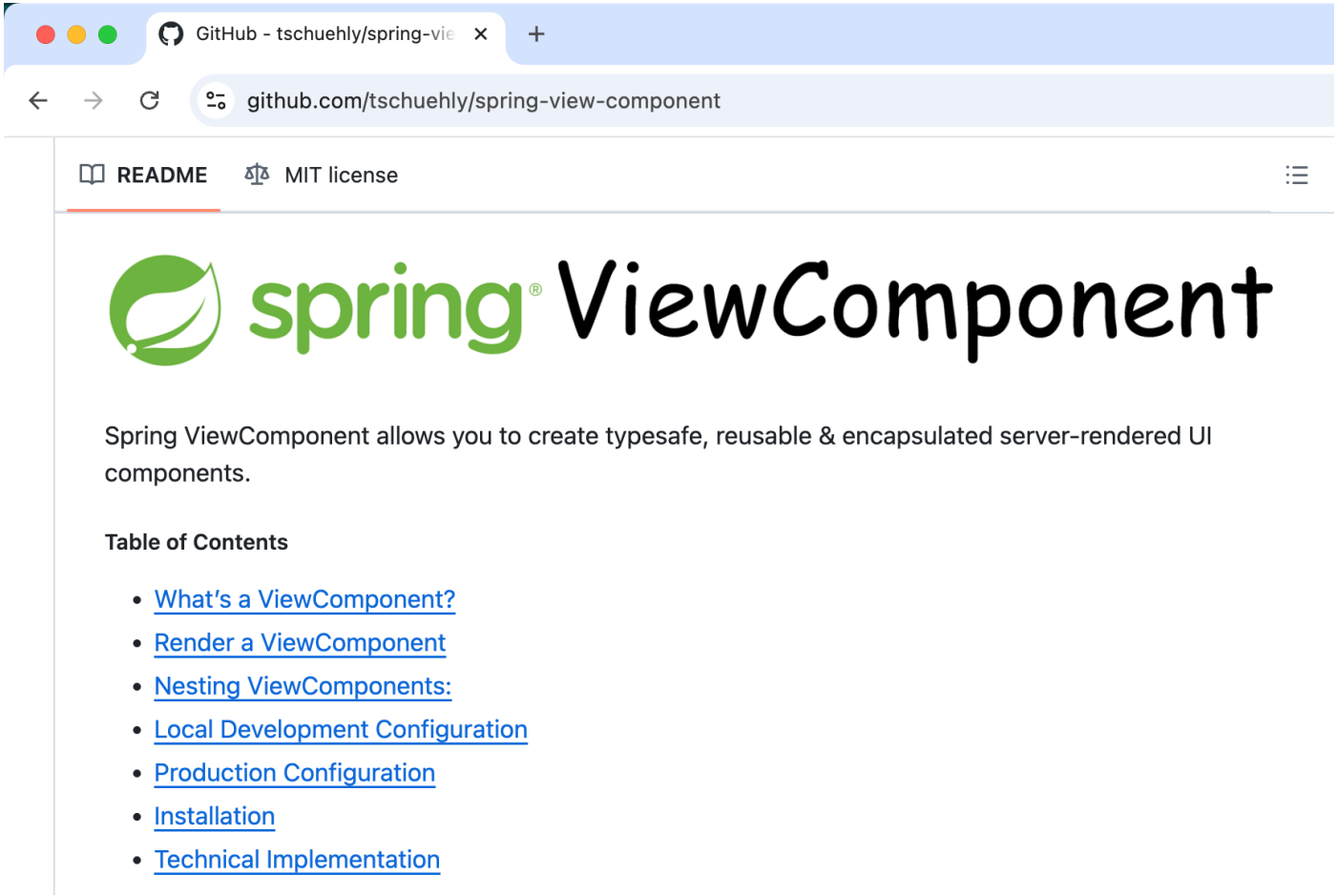


UserManagement.jte

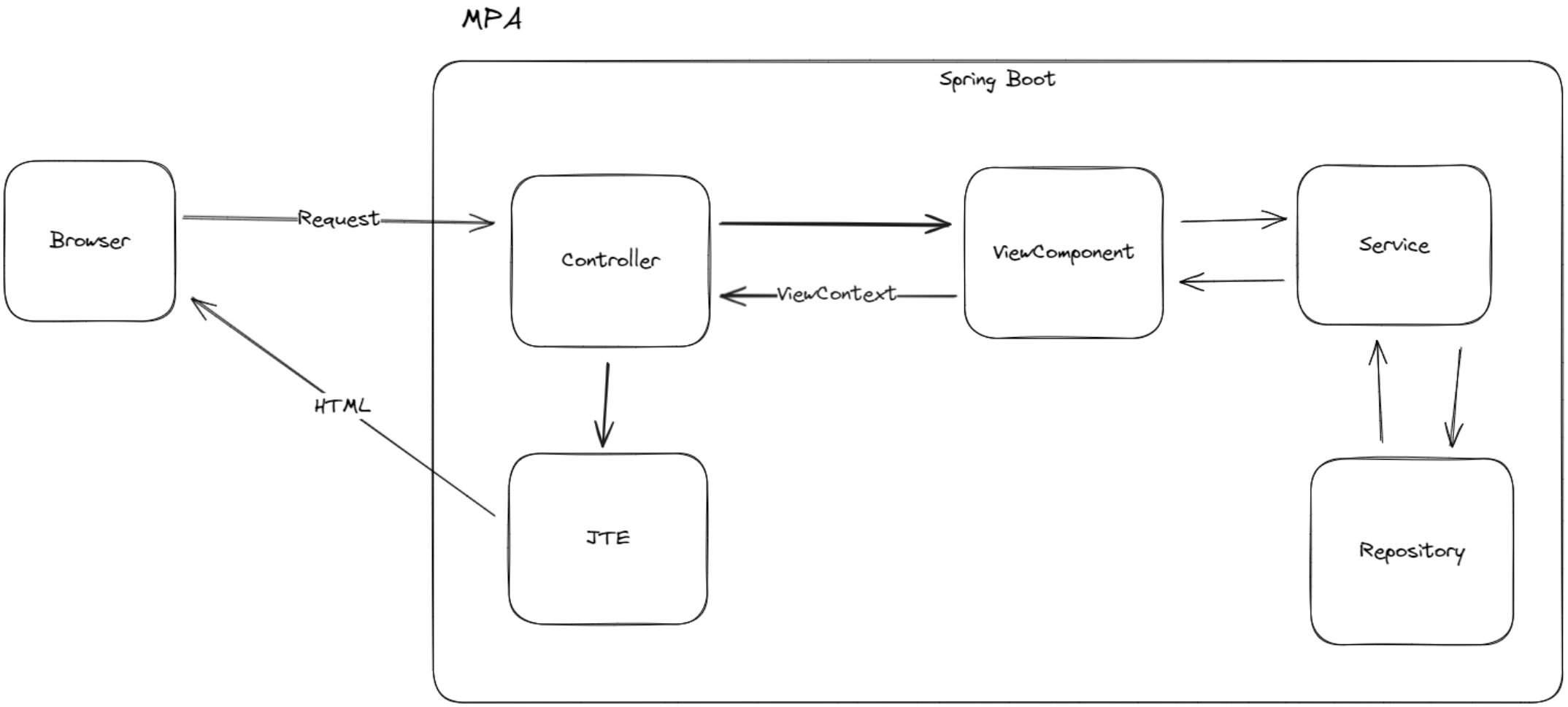
```
@import de.tschuehly.easy.spring.auth.user.EasyUser
@param java.util.List<EasyUser> easyUserList

@for(var user: easyUserList)
    ${user.username}
@endfor
```


15. Web MVC framework
https://docs.spring.io/spring-framework/docs/3.0.0.RELEASE/reference/html/mvc.html#mvc-introduction
@Controller
public class HelloWorldController {
@RequestMapping("/helloWorld")
public ModelAndView helloWorld() {
ModelAndView mav = new ModelAndView();
mav.setViewName("helloWorld");
mav.addObject("message", "Hello World!");
return mav;
}
}
SPRING FRAMEWORK CHANGELOG
==========================
http://www.springsource.org
Changes in version 3.0.0.GA (2009-12-16)

ViewComponent



Star it on GitHub!


MPA
Spring Boot
Browser
Request
Controller
ViewComponent
Service
ViewContext
HTML
JTE
Repository

UserManagement.java

```java
@ViewComponent
public class UserManagement {
  private final UserService userService;

  public UserManagement(UserService userService) {
    this.userService = userService;
  }

  public record UserManagementContext(java.util.List<EasyUser> userTable)
        implements ViewContext{}

  public ViewContext render(){
    return new UserManagementContext(userService.findAll());
  }
}
```

UserManagement.java

```java
@ViewComponent
public class UserManagement {
  private final UserService userService;

  public UserManagement(UserService userService) {
    this.userService = userService;
  }

  public record UserManagementContext(java.util.List<EasyUser> userTable)
        implements ViewContext{}

  public ViewContext render(){
    return new UserManagementContext(userService.findAll());
  }
}
```

UserManagement.java

```java
@ViewComponent
public class UserManagement {

  public record UserManagementContext(java.util.List<EasyUser> userTable)
        implements ViewContext{}

  public ViewContext render(){
    return new UserManagementContext(userService.findAll());
  }
}
```

UserManagement.jte

```
@import de.tschuehly.easy.spring.auth.user.management.UserManagement.UserManagementContext
@param UserManagementContext userManagementContext
@for(var user: userManagementContext.easyUserList())
    ${user.username}
@endfor
```

UserController.java

```java
@Controller
public class UserController {

  private final UserManagement userManagement;

  public UserController(UserManagement userManagement) {
    this.userManagement = userManagement;
  }

  @GetMapping("/")
  public ViewContext index() {
    return userManagement.render();
  }
}
```

UserController.java

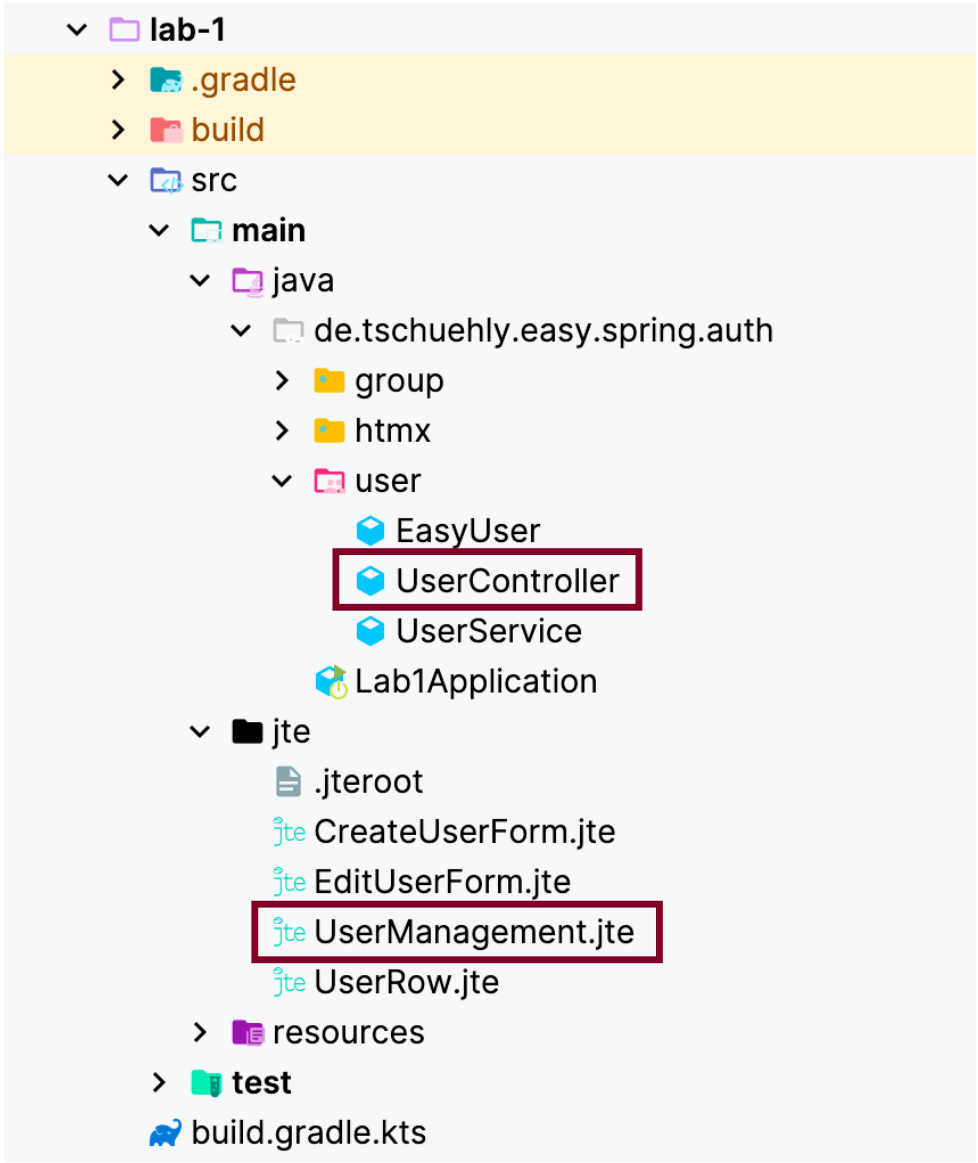
```java
@Controller
public class UserController {

  private final UserManagement userManagement;

  public UserController(UserManagement userManagement) {
    this.userManagement = userManagement;
  }

  @GetMapping("/")
  public ViewContext index() {
    return userManagement.render();
  }
}
```

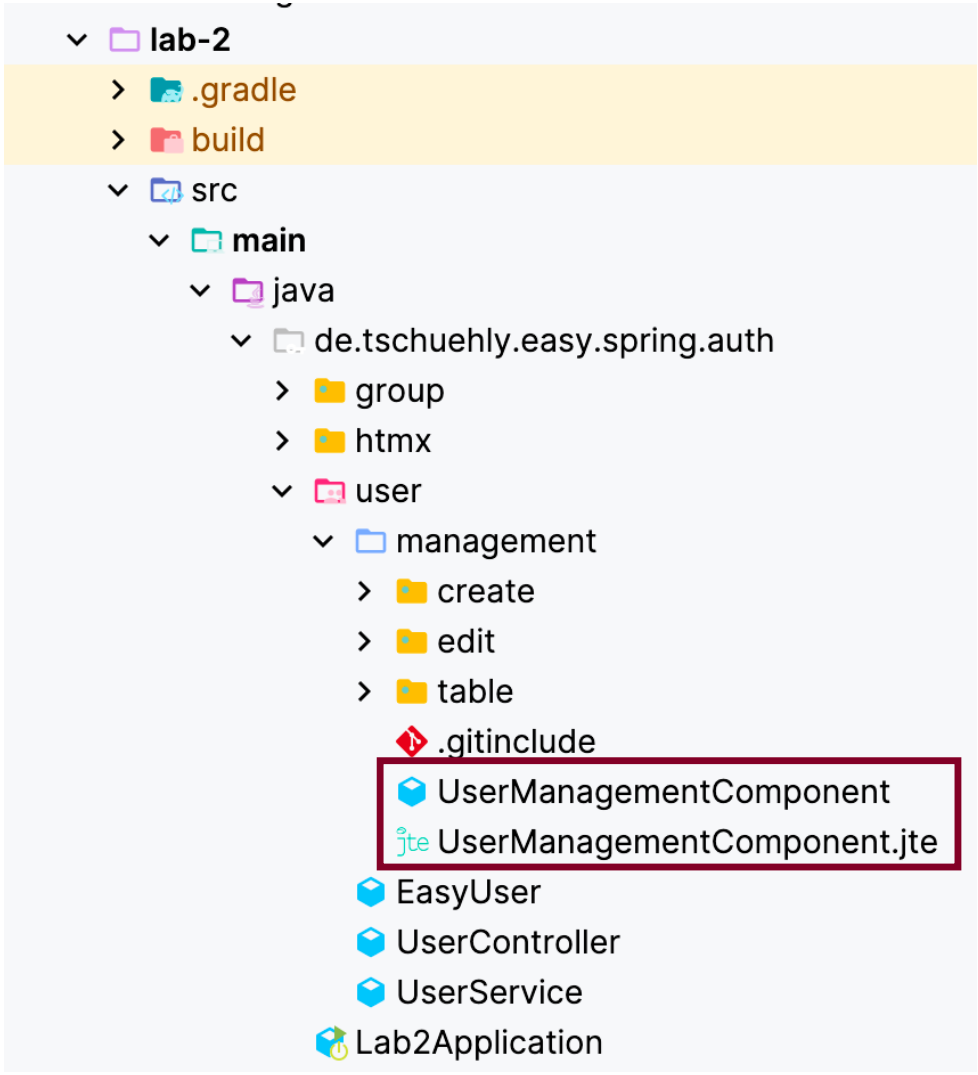
LayoutViewComponent.java

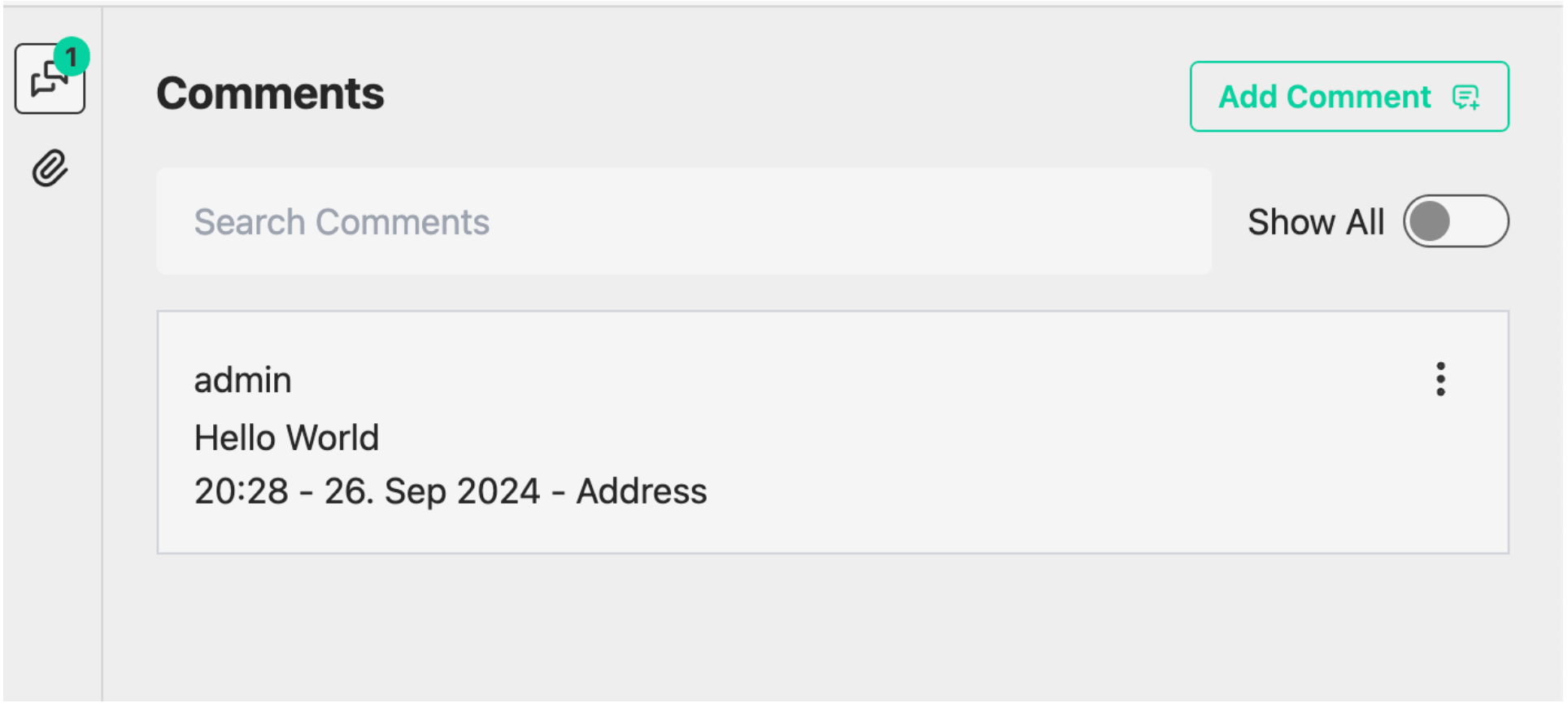
```java
@ViewComponent
public
class LayoutViewComponent {
  private record LayoutView(ViewContext nestedViewComponent) implements ViewContext {}

  public ViewContext render(ViewContext nestedViewComponent) {
    return new LayoutView(nestedViewComponent);
  }
}
```

LayoutViewComponent.jte

```
@param layoutView:
de.tschuehly.kteviewcomponentexample.web.layout.LayoutViewComponent.LayoutView
<nav>
  This is a Navbar
</nav>
<body>
 ${layoutView.nestedViewComponent}
</body>
<footer>
  This is a footer
</footer>
```


lab-1
.gradle
build
src
main
java
de.tschuehly.easy.spring.auth
group
htmx
user
EasyUser
UserController
UserService
Lab1Application
jte
.jteroot
CreateUserForm.jte
EditUserForm.jte
UserManagement.jte
UserRow.jte
resources
test
build.gradle.kts


lab-2
.gradle
build
src
main
java
de.tschuehly.easy.spring.auth
group
htmx
user
management
create
edit
table
.gitinclude
UserManagementComponent
UserManagementComponent.jte
EasyUser
UserController
UserService
Lab2Application


1
Comments
Add Comment
Search Comments
Show All
admin
Hello World
20:28 - 26. Sep 2024 - Address

```
<form hx-post="${commentAreaContext.createComment()}"
      hx-on:htmx:after-request="this.reset();">
  <textarea name="commentText"></textarea>
  <button type="submit">
    Create Comment
  </button>
</form>
<div id="${CommentAreaContext.commentListId}">
</div>
```

```java
public record CommentAreaContext(
    CommentAreaDefinition commentAreaDefinition
  ) implements ViewContext {

  public static final String CREATE_COMMENT = "/api/comment";

  public Url createComment() {
    return Url.path(CREATE_COMMENT)
        .addParam("dataItemId", commentAreaDefinition.dataItemId());
  }
}
```

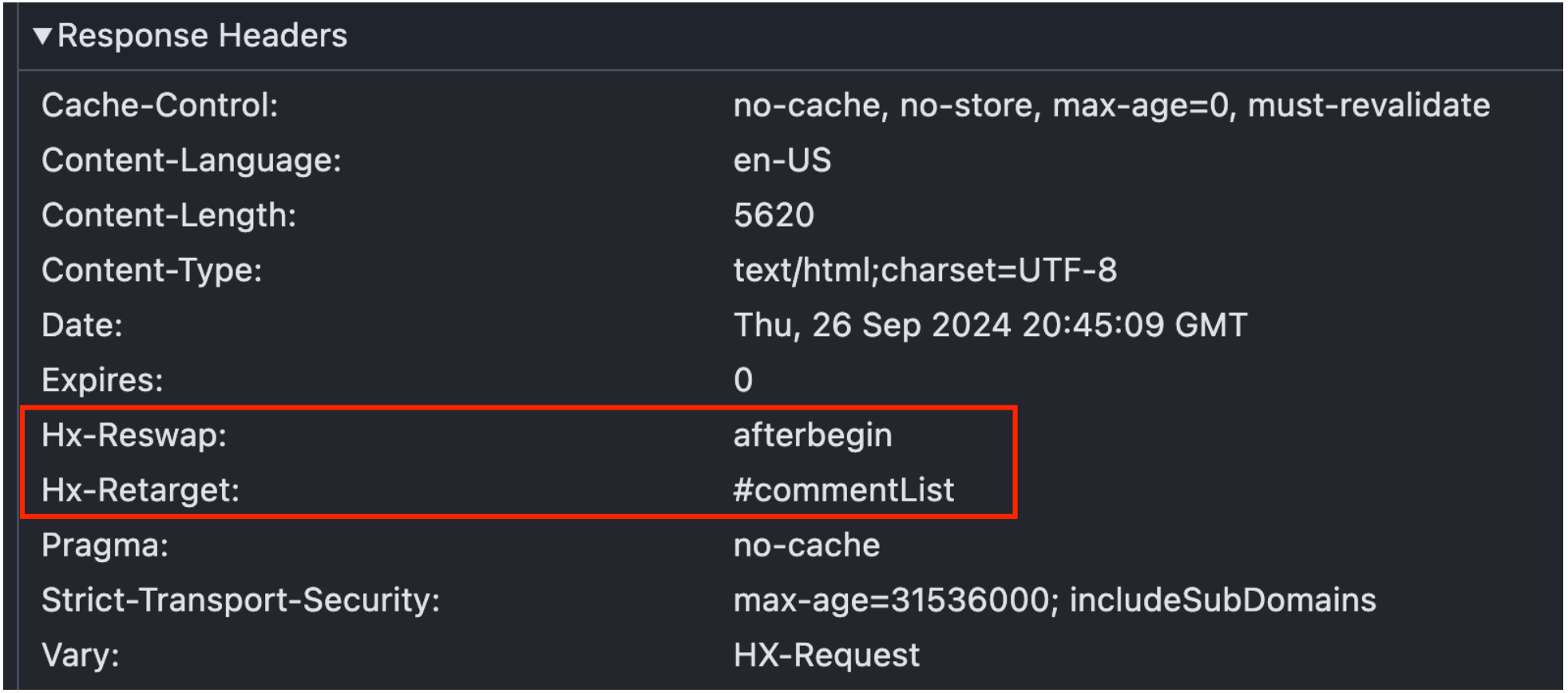
```
public class Url implements Content {

  private UriComponentsBuilder uriComponentsBuilder;

  @Override
  public void writeTo(TemplateOutput output) {
    output.writeContent(uriComponentsBuilder.build().toString());
  }
}
```

```
<form hx-post="/api/comment?dataItemId=5afbd5e5-ec95-468f-81d8-1612da3a070e"
</form>
```

```
public Url addParam(String name, Object value) {
  if (Objects.isNull(value)) {
    return this;
  }
  uriComponentsBuilder.queryParam(name, URLEncoder.encode(
    value.toString(), StandardCharsets.UTF_8));
  return this;
}
```

```java
@ViewComponent
@Controller
public class CommentAreaComponent {

  @PostMapping(CREATE_COMMENT)
  public ViewContext createComment(
      @RequestParam UUID dataItemId,
      @RequestParam String commentText
  ) {
    Comment comment = commentService.createComment(dataItemId, commentText);
    HtmxUtil.retargetToId(CommentAreaContext.commentListId);
    HtmxUtil.reswap(HxSwapType.AFTER_BEGIN);
    return commentComponent.render(new CommentDefinition(comment));
  }
}
```

```java
public class HtmxUtil {

  public static void retargetToId(String id) {
    setHeader(HtmxResponseHeader.HX_RETARGET, "#" + id);
  }

  public static void reswap(HxSwapType hxSwapType) {
    setHeader(HtmxResponseHeader.HX_RESWAP, hxSwapType.getValue());
  }
}
```


▼Response Headers
Cache-Control: no-cache, no-store, max-age=0, must-revalidate
Content-Language: en-US
Content-Length: 5620
Content-Type: text/html;charset=UTF-8
Date: Thu, 26 Sep 2024 20:45:09 GMT
Expires: 0
Hx-Reswap: afterbegin
Hx-Retarget: #commentList
Pragma: no-cache
Strict-Transport-Security: max-age=31536000; includeSubDomains
Vary: HX-Request

htmx.tschuehly.de

Building server-side web applications with htmx

User Management Group Management Workshop GitHub Repository

> Spring is typically only used for JSON API backend development, while the web frontend is built with a JavaScript framework.
> htmx enables us to create interactive web applications with server-side rendered templates without JavaScript.
> In my workshop, you will learn to use patterns you know and love from building backends to create the dynamic full-stack application you see here.

Search Users

uuid	username	password	
31fa0ad0-8c21-4b1d-a90c-43a9776f0870	keneth.wolf	7cwsau4mb5dg	
1e9fdbde-1637-4a54-8fb2-d0df8e89e332	jay.lind	6i79p20ksamtyd2	
4a474c1c-4850-40f9-b490-82c5d27dca78	isreal.bergstrom	3r1sg6k5g0	
694c9d07-e17e-4572-be1e-0abb4d62048f	Thomas	This is a password	

Inline Editing

http://127.0.0.1:8081/group-management

Using Spring Beans to compose the UI

http://127.0.0.1:8082/group-management

Using Spring Beans to compose the UI

http://127.0.0.1:8086/

AI-Agents lieben htmx + Spring Boot

- Ein Tech-Stack, eine Sprache = weniger Kontext für den Agent
- Kein Frontend/Backend-Kontextwechsel
- Templates sind einfaches HTML mit Attributen

AI-Agents lieben htmx + Spring Boot

- ViewComponents = in sich geschlossene Einheiten
- Typsicherheit durch JTE → Compiler fängt Fehler, nicht der Agent
- Locality of Behaviour → für Mensch UND Maschine

htmx 4.0 - The fetch()ening

- fetch() statt XMLHttpRequest → ermöglicht Streaming & SSE im Core
- Built-in Morphing (idiomorph) für intelligente DOM-Updates
- View Transitions API für sanfte Animationen
- <htmx-partial> Tag für Multi-Target-Swaps
- hx-get, hx-post, hx-target, hx-swap → bleiben gleich!

Warum htmx + Spring Boot

- Weniger Komplexität: Kein Client-Side State, kein API-Layer
- Schnellere Entwicklung: Ein Stack, eine Sprache, ein Deployment
- Bessere Wartbarkeit: Locality of Behaviour
- Perfekt für AI Agents: Einfacher Kontext, Typsicherheit
- Zukunftssicher: htmx 4.0, wachsende Community

Thank you for listening!

tschuehly.de
github.com/tschuehly
x.com/tschuehly
linkedin.com/in/tschuehly

Function*

Mustermann

Authority*

Landratsamt Musterstadt

Firstname*

Max

Lastname*

Mustermann

E-Mail Address*

max.mustermann@example.co

Phone Number*

+491234567890

Address

Street Name*

Musterstraße

Street Number*

1

Unit Number*

12345

Postal Code*

98765

City*

Musterstadt

State*

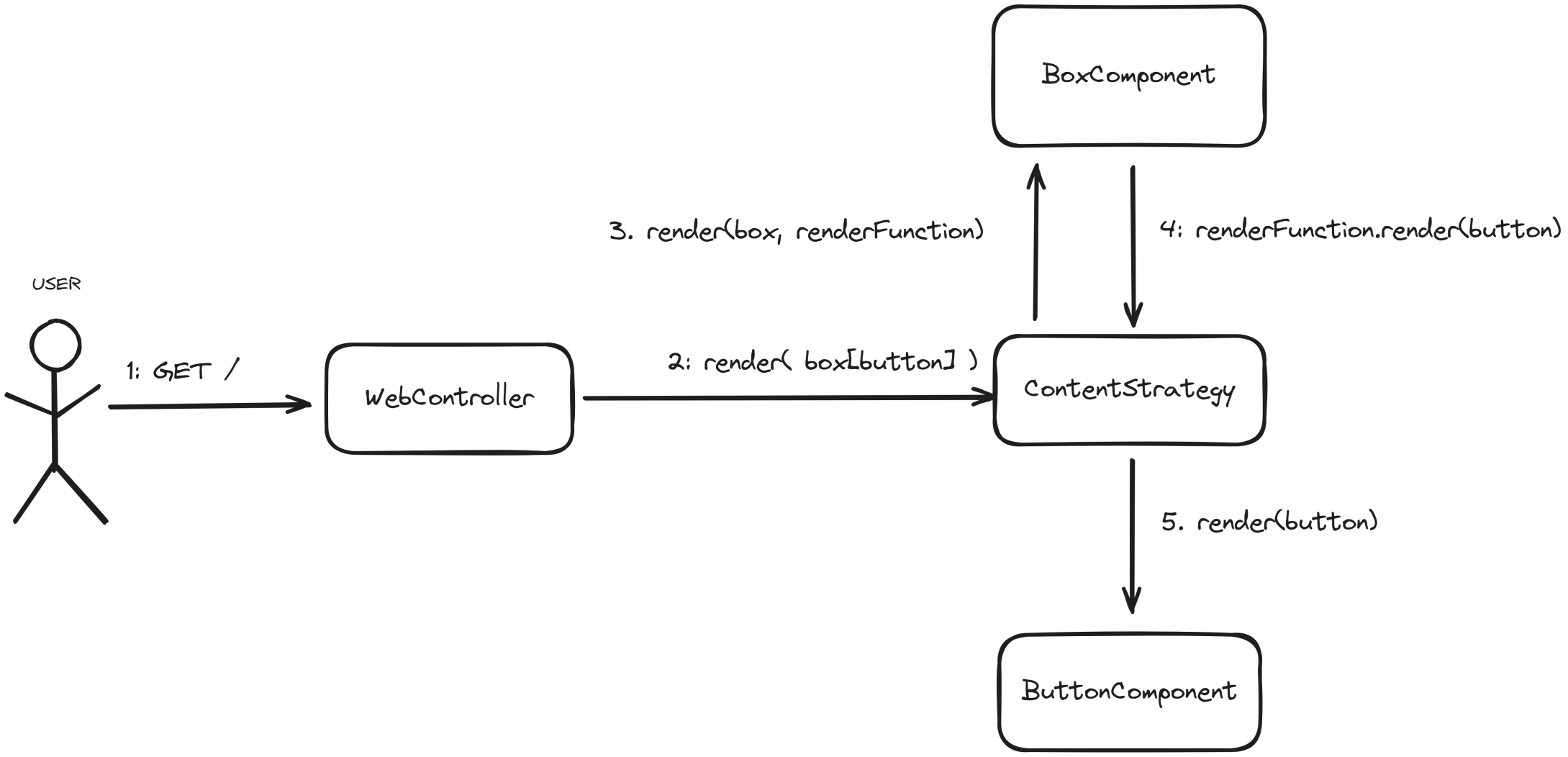
Musterstaat

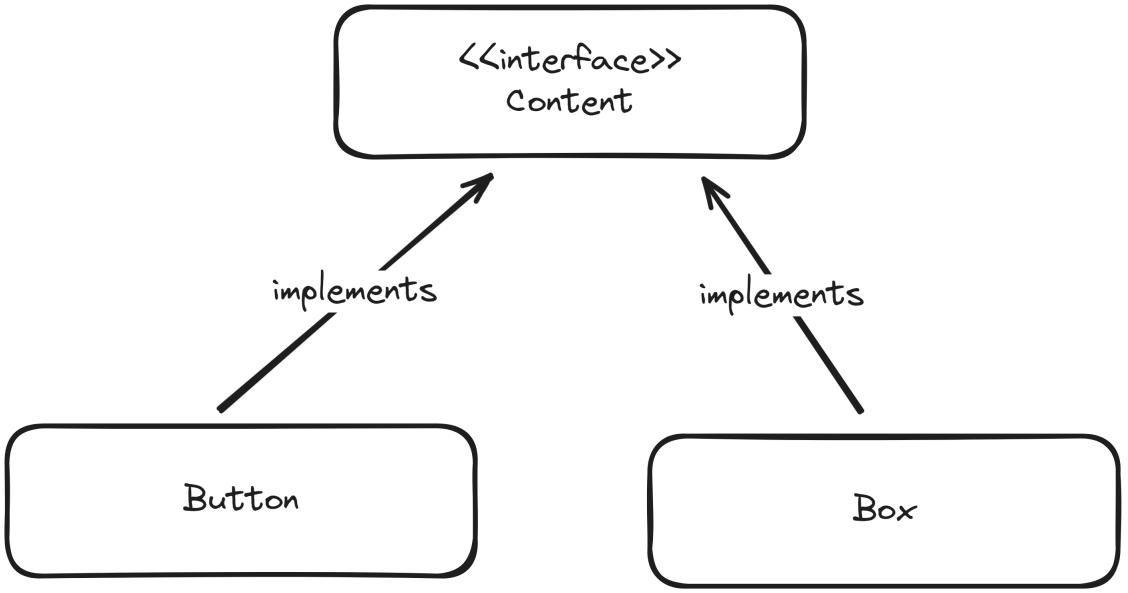
Country*

Musterland

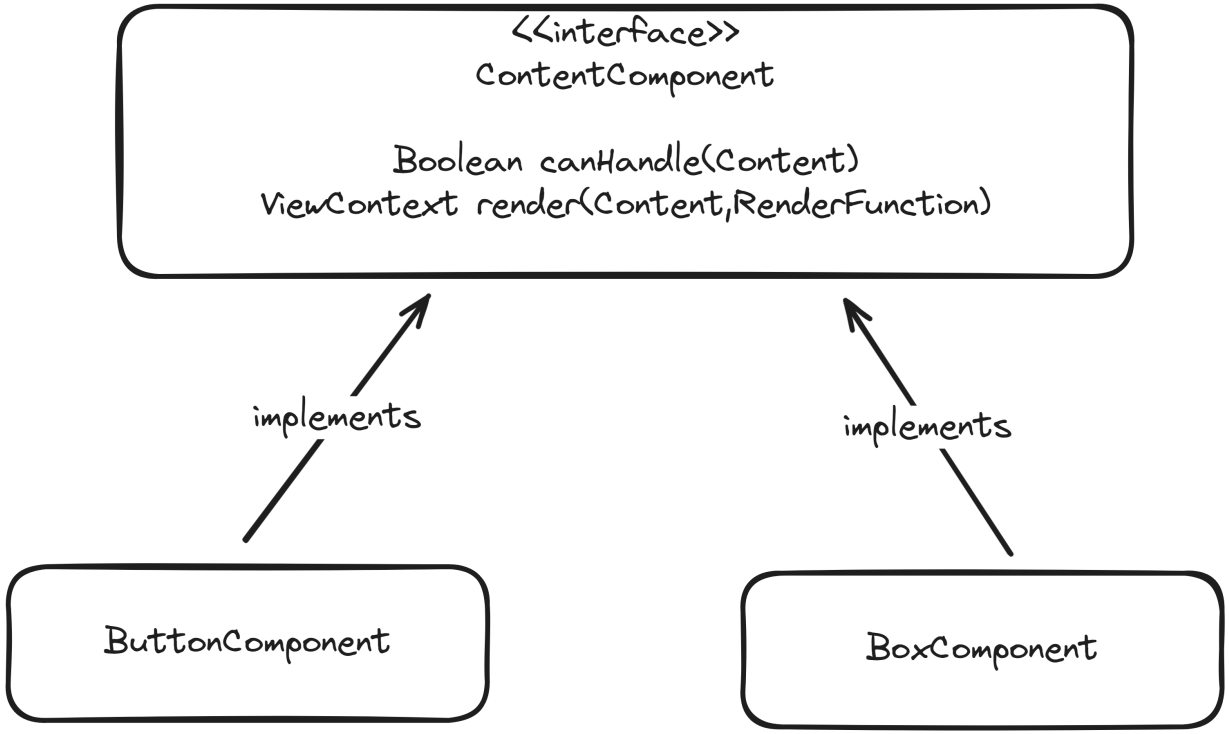
Complete

```java
@Controller
public class WebController {

  @Autowired
  private ContentStrategy contentStrategy;

  @GetMapping("/")
  ViewContext index(){
    return contentStrategy.render(
        new Box(
            new Button("Hello World")
        )
    );
  }
}
```


BoxComponent
3. render(box, renderFunction)
4: renderFunction.render(button)
USER
1: GET /
WebController
2: render(box[button])
ContentStrategy
5. render(button)
ButtonComponent



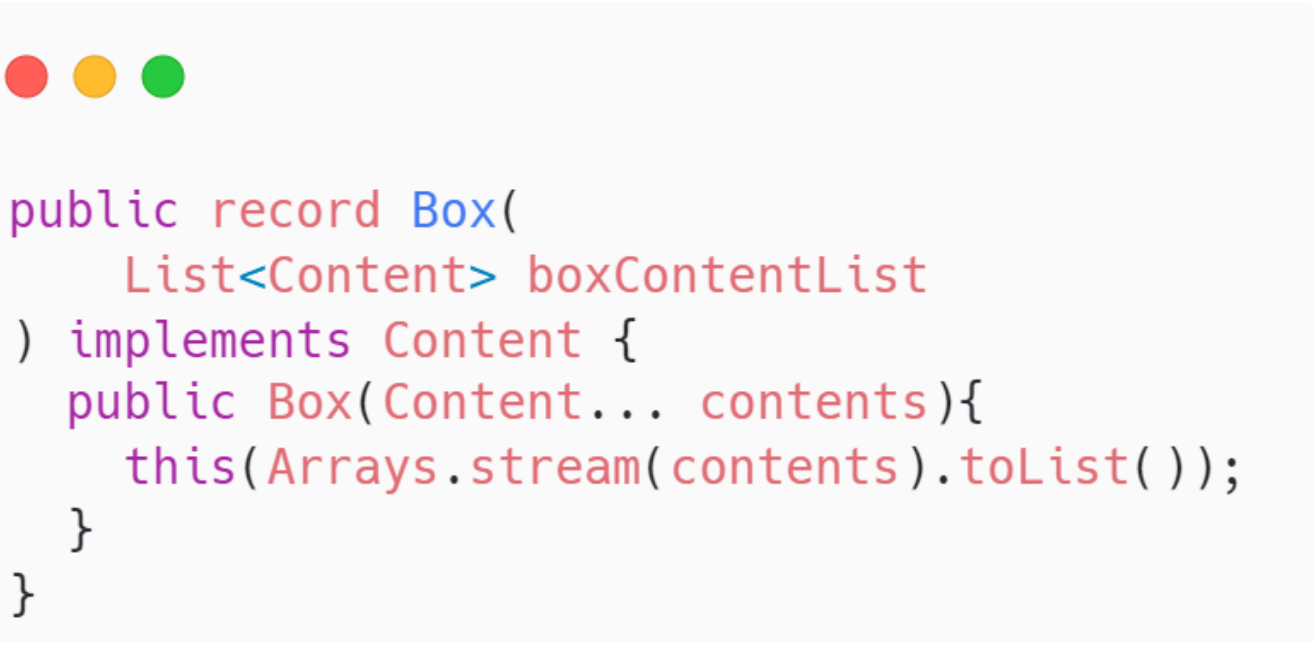

```java
public record Button(String label) implements Content {

}
```


<<interface>>
ContentComponent
Boolean canHandle(Content)
ViewContext render(Content,RenderFunction)
implements
implements
ButtonComponent
BoxComponent

```java
@ViewComponent
public class ButtonComponent implements ContentComponent {

  @Override
  public Boolean canHandle(Content content) {
    return content instanceof Button;
  }

  @Override
  public ViewContext render(
    Content content,
    RenderFunction renderFunction
  ) {
    return new ButtonContext((Button) content);
  }


  public record ButtonContext(Button button)
    implements ViewContext {}
}
```

```
@param de.tschuehly.svc.ui.field.button.ButtonComponent.ButtonContext buttonContext

<button>
    ${buttonContext.button().label()}
</button>
```

```java
@Service
public class ContentStrategy {

  @Autowired
  private final List<ContentComponent> contentComponents;

  public ViewContext render(Content content) {
    ContentComponent contentComponent = contentComponents.stream()
        .filter(it -> it.canHandle(content))
        .findFirst()
        .orElseThrow(() -> new RuntimeException("FormNotImplementedException"));
    return contentComponent.render(content, this::render);
  }
}
```



```java
public record Box(
    List<Content> boxContentList
) implements Content {
  public Box(Content... contents){
    this(Arrays.stream(contents).toList());
  }
}
```

```java
@ViewComponent
public class BoxComponent implements ContentComponent {
  @Override
  public Boolean canHandle(Content content) {
    return content instanceof Box;
  }

  @Override
  public ViewContext render(Content content, RenderFunction renderFunction) {
    return new BoxContext((Box) content, renderFunction);
  }
  public record BoxContext(Box content, RenderFunction renderFunction) implements ViewContext {}
}
```

```
@param de.tschuehly.svc.ui.layout.box.BoxComponent.BoxContext boxContext

<div>
    @for(var content: boxContext.content().boxContentList())
        ${boxContext.renderFunction().render(content)}
    @endfor
</div>
```

```java
@Controller
public class WebController {

  @Autowired
  private ContentStrategy contentStrategy;

  @GetMapping("/")
  ViewContext index(){
    return contentStrategy.render(
        new Box(
            new Button("Hello World")
        )
    );
  }
}
```