boliga.dk performance

express middleware

- each request is run through ~200 regex tests
- not needed for static files
- not needed for valid urls

quick fix

- load middleware after static files

complete fix

- remove middleware
- extend angular 404 handler to test url

```
app.get('/app/info', (req: express.Request, res: express.Response) => {
  res.status(200).send({ url: environment.apiUrl });
});
// SEO sitemap
app.get('/sitemap/*', sitemap);
// custom landingPages
app.get('/boligadvokatkontoret', staticBoligadvokatkontoret );

// Server static files from /browser
if (appConfig.REQUESTLOG === 'all') {
  app.use(morgan(morganJsonFormat, morganConfig(appConfig.REQUESTLOGMINSTATUS)))
}
app.get('*.*', express.static(join(appConfig.CURR_FOLDER, 'browser')));

// Setup for handling non-static routes
app.engine('html', ngExpressEngine({
  bootstrap: AppServerModule
}));
app.set('view engine', 'html');
app.set('views', join(appConfig.CURR_FOLDER, 'browser'));
app.use(guidRedirecthandling());
app.use(newsStatsDataRedirecthandling());
app.use(redirecthandling(redirectRules));
app.use(lowercaseRedirecting());
app.use(redirectExcludedQueryParams());
app.use(NotFoundSeoRedirectHandling());

// All regular routes use the Universal engine
if (appConfig.REQUESTLOG === 'routes') {
  app.use(morgan(morganJsonFormat, morganConfig(appConfig.REQUESTLOGMINSTATUS)))
}
app.get('*', (req, res) => {
  res.render(join(appConfig.CURR_FOLDER, 'browser', 'index.html'), {
    req,
    res,
    providers: [
      {
        provide: 'REQUEST', useValue: (req)
      },
      {
        provide: 'RESPONSE', useValue: (res)
      }
    ],
  });
});
```

render blocking

- each script and css file in <head> blocks the page display
- external scripts & css moved to body with async tags
- preconnect tags added for API & image CDN

items moved to async (non blocking) loading:

- google tag manager
- facebook sdk
- cookie consent
- google fonts

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="description" content="">
  <meta name="keywords" content="">
  <title>Boliga</title>
  <base href="/">
  <meta name="viewport" content="width=device-width, initial-scale=
  <link rel="icon" type="image/x-icon" href="favicon.ico">
  <link rel="apple-touch-icon" sizes="180x180" href="/assets/apple-
  <link rel="icon" type="image/png" sizes="32x32" href="/assets/fav
  <link rel="icon" type="image/png" sizes="16x16" href="/assets/fav
  <link rel="manifest" href="/assets/site.webmanifest">
  <link rel="mask-icon" href="/assets/safari-pinned-tab.svg" color=
  <meta name="msapplication-TileColor" content="#da532c">
  <meta name="theme-color" content="#ffffff">
  <link rel="preconnect" href="https://api.boliga.dk">
  <link rel="preconnect" href="https://i.boliga.org">
  <!--[if IE]>
  <link href="https://fonts.googleapis.com/css?family=Montserrat:40
  <![endif]-->
</head>
<body>
  <app-root></app-root>
  <script>(function(w,d,s,l,i){w[l]=w[l]||[];w[l].push({'gtm.start'
    new Date().getTime(),event:'gtm.js'});var f=d.getElementsByTagN
    j=d.createElement(s),dl=l!='dataLayer'?'&l='+l:'';j.async=true;
    'https://www.googletagmanager.com/gtm.js?id='+i+dl;f.parentNode
    })(window,document,'script','dataLayer','GTM-PWD5VZT');
  </script>
  <script type="text/javascript" src="https://connect.facebook.net/
  <script id="CookieConsent" src="https://policy.app.cookieinformat
  <!--[if !IE]> -->
  <link href="https://fonts.googleapis.com/css?family=Montserrat:40
  <!-- <![endif]-->
</body>
</html>
```

font fallback

- no text displayed until fonts downloaded
- poor experience on slow connections
- pages fully visible with no text

fixed with fallback font display

- font-display: swap
- text visible while font is downloading
- once font is downloaded displayed font is updated

```
  font-family: 'PT Sans';
  src: local('PT Sans Bold'), local('PTSans-Bold'), url('fonts/PTSans-Bold.tt
  font-weight: bold;
  font-style: normal;
  font-display: swap;
}
@font-face {
  font-family: 'PT Sans';
  src: local('PT Sans Italic'), local('PTSans-Italic'), url('fonts/PTSans-Ita
  font-weight: normal;
  font-style: italic;
  font-display: swap;
}
@font-face {
  font-family: 'PT Sans';
  src: local('PT Sans Bold Italic'), local('PTSans-BoldItalic'), url('fonts/P
  font-weight: bold;
  font-style: italic;
  font-display: swap;
}
@font-face {
  font-family: 'PT Sans';
  src: local('PT Sans'), local('PTSans-Regular'), url('fonts/PTSans-Regular.t
  font-weight: normal;
  font-style: normal;
  font-display: swap;
}
/* Google Fonts - Libre Baskerville */
@font-face {
  font-family: 'Libre Baskerville';
  src: local('Libre Baskerville Bold'), local('LibreBaskerville-Bold'), url('
  font-weight: bold;
  font-style: normal;
  font-display: swap;
}
@font-face {
  font-family: 'Libre Baskerville';
  src: local('Libre Baskerville'), local('LibreBaskerville-Regular'), url('fo
  font-weight: normal;
  font-style: normal;
  font-display: swap;
}
@font-face {
  font-family: 'Libre Baskerville';
  src: local('Libre Baskerville Italic'), local('LibreBaskerville-Italic'), u
```

double startup

- loading app.module twice
- does not seem to be needed
- helps decrease mobile startup time

```
import { enableProdMode } from '@angular/core';
import { platformBrowserDynamic } from '@angular/platform-br

import { AppModule } from './app/app.module';
import { environment } from './environments/environment';

if (environment.production) {
  enableProdMode();
}

platformBrowserDynamic().bootstrapModule(AppModule)
  .catch(err => console.log(err));


// !!!! this triggers a full reload of './app/app.module' !
// document.addEventListener('DOMContentLoaded', () => {
//   platformBrowserDynamic()
//     .bootstrapModule(AppModule)
//     .catch(err => console.log(err));
// });
```

double nav bar

- two nav bars blue background & transparent background
- both layouts loaded for every single page
- only difference is a single css class

fixed with dynamic css class

- removed transparent nav bar references
- uses angular router data attribute
- active on routes that need transparent background

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';

@Component({
  selector: 'app-main-layout',
  templateUrl: './main-layout.component.html',
  styleUrls: ['./main-layout.component.scss']
})
export class MainLayoutComponent implements OnInit {

  transparent: boolean = false;

  constructor(private route: ActivatedRoute) { }

  ngOnInit(): void {
    this.transparent = this.route.snapshot.data['isHomepage'] || fals
  }
}
```

heavy app.module

- too many things loaded into main bundle
- does not use lazy loading
- needs to be as small as possible

quick fix

- comment out stuff that does not seem to be used
- load services/modules at component level when needed

complete fix

- review everything here and remove as much as possible

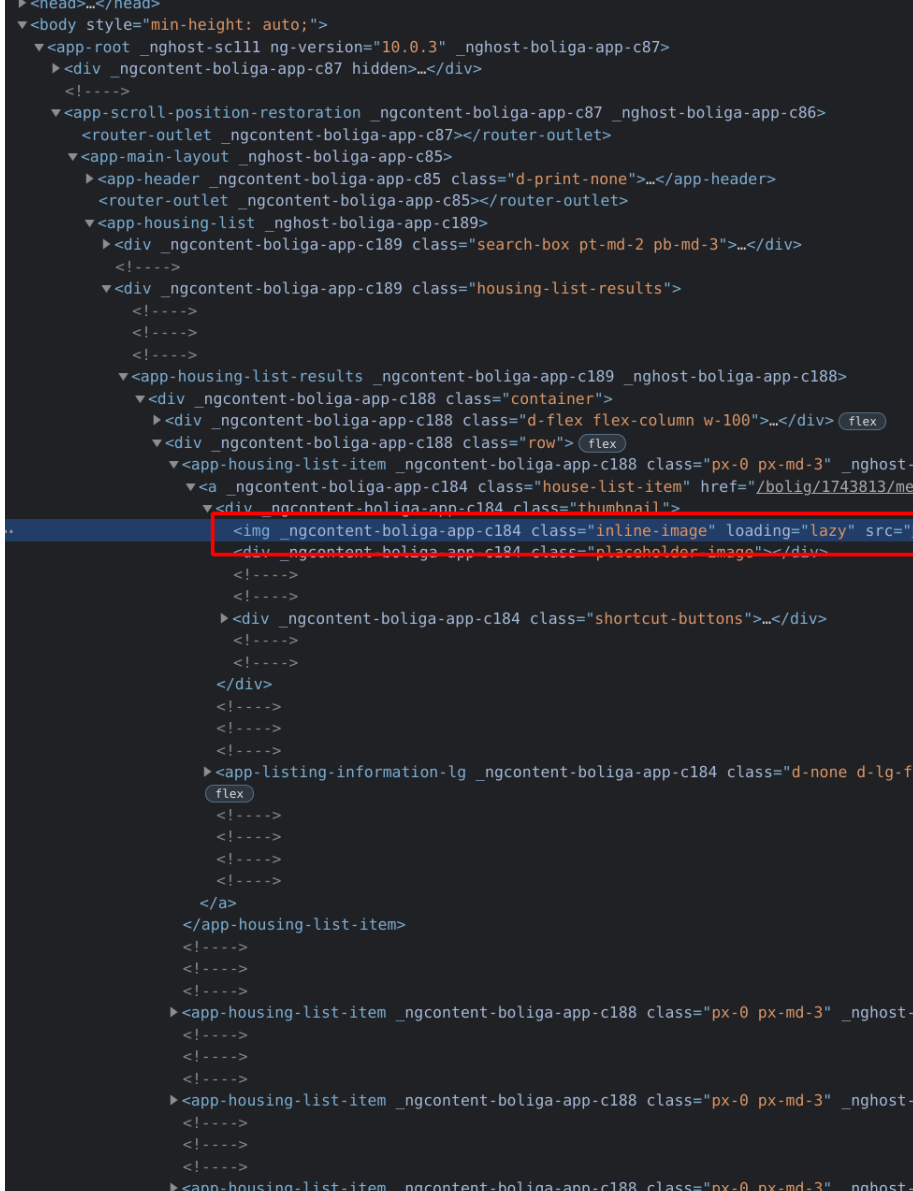
```
import { BrowserModule, BrowserTransferStateModule, TransferState, Ha
import { BrowserAnimationsModule } from '@angular/platform-browser/an
import { APP_ID, APP_INITIALIZER, ErrorHandler, Inject, Injector, LOC
import { isPlatformBrowser, LOCATION_INITIALIZED, registerLocaleData
import { MAT_DATE_LOCALE } from '@angular/material/core';
import { MAT_TOOLTIP_DEFAULT_OPTIONS, MatTooltipDefaultOptions } fro
import { HTTP_INTERCEPTORS, HttpClient, HttpClientModule } from '@ang
import { TranslateLoader, TranslateModule, TranslateService } from '@
import { TransferHttpCacheModule } from '@nguniversal/common';
import { InlineSVGModule } from 'ng-inline-svg';
import { CookieService } from 'ngx-cookie-service';
import { FacebookModule } from 'ngx-facebook';

import { TranslateBrowserLoader } from './core/translate/translate-br
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { AuthModule, PreRequestAuthInterceptor } from './core/auth';
import { LayoutModule } from './core/layout/layout.module';
// import { SearchModule } from './modules/search/search.module';
// import { SearchAutocompleteModule } from './modules/location-searc
// import { PopularCategoriesModule } from './modules/popular-categor
// import { BriefSiteStatisticsModule } from './modules/brief-site-st
// import { LoginModalModule } from './modules/login-modal/login-moda
import { GlobalErrorService } from './helper/global-error.service';

import da from '@angular/common/locales/da';
import { LinkService } from './helper/link.service';
import { LoggerModule, NgxLoggerLevel } from 'ngx-logger';
import { DEFAULT_TIMEOUT, TimeoutInterceptor, TIMEOUT_RULES } from '
import { SeoService } from './helper/seo.service';
// import { MapHelperService } from './helper/map-helpers.service';
import { GtmService } from './helper/gtm/gtm.service';
import { DataLayerService } from './helper/gtm/data-layer.service';
import { ListingDetailsService } from './helper/listing-details.servi
import { UtmService } from './helper/utm/utm.service';
import { Observable, combineLatest } from 'rxjs';
// import { HTMLMarkerPopupComponent } from './modules/visual-search/
import { ScrollPositionRestorationModule } from './core/scroll-positi
import { ServiceWorkerModule } from '@angular/service-worker';
import { environment } from '../environments/environment';
```

lazy image loading

- page does not wait for images to become 'ready'
- visible on screen images loaded first
- page scroll loads off screen images on demand
- built into browser - no script overhead

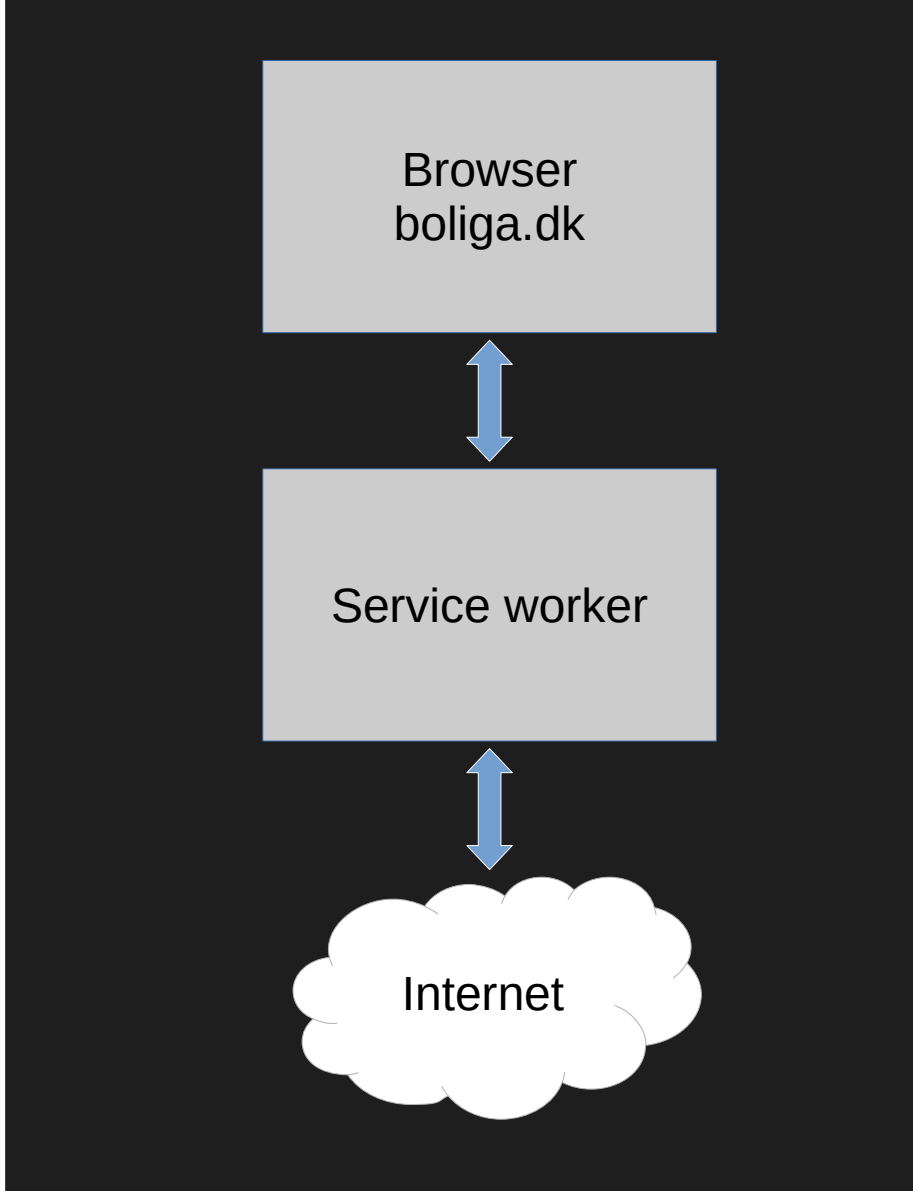
important:

- only works with <img> tag
- CSS background image not supported



service worker

- separate background worker
- handles advanced caching strategies
- list of files to cache generated at build time
- cache gets busted for changed files on deploy
- assists in caching external content (e.g. google fonts)
- built in failsafe - automatic uninstall
- instant network responses for slow mobile connections



change of design

- two large application design elements missing
- causing heavy slowdown in responses

server vs browser code split

- running unnecessary code on the server

server state transfer

- state is lost after SSR complete

server vs browser

- running unnecessary code on the server
- interactive elements (buttons, maps, dynamic ui elements, etc)
- global code hooks (browser scroll tracking, analytics, facebook, etc)

example; inspiration button

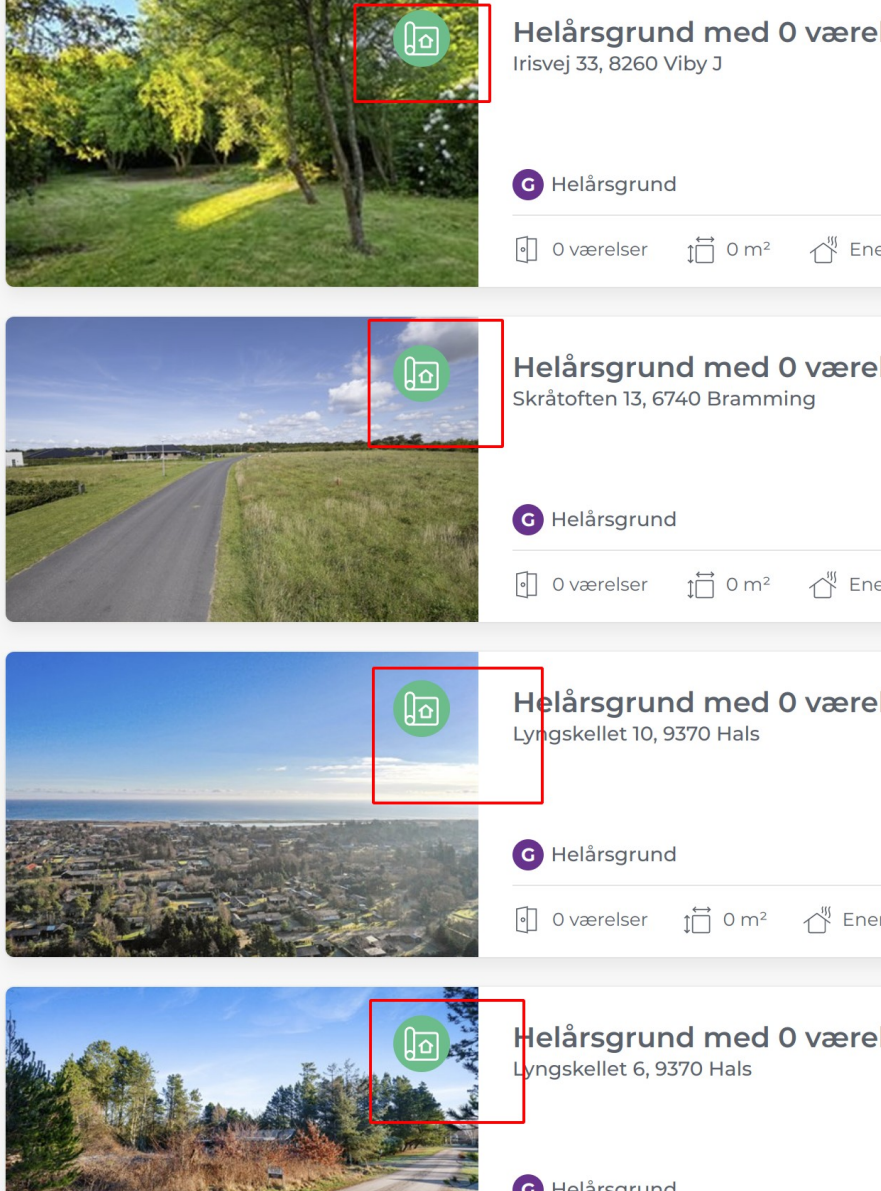
- extremely expensive to create
- triggers many API requests for results screen
- the server will not click the button
- google will not click the button
- only in the browser maybe the button is clicked

quick fix

- changed this button to only render on browser

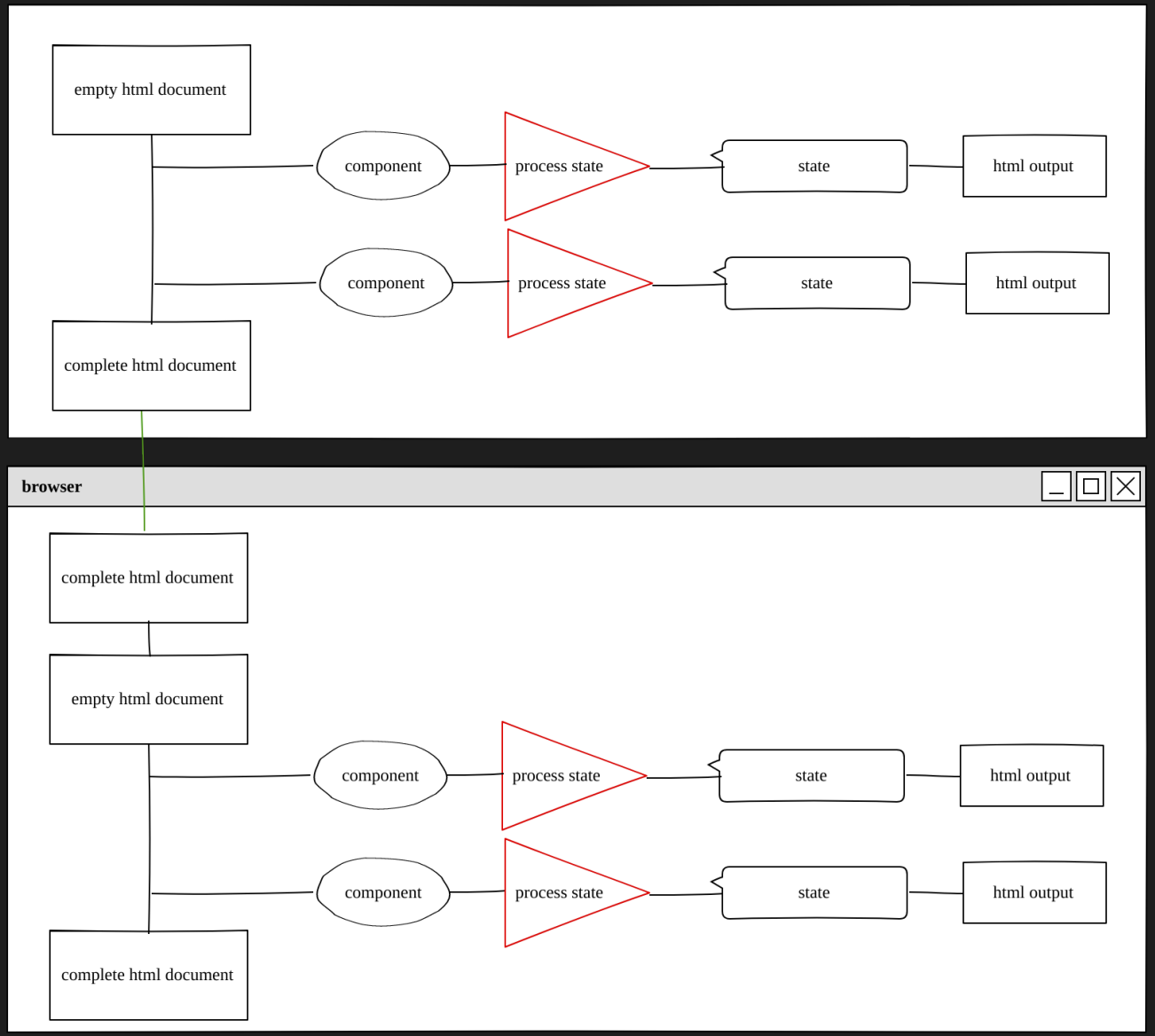
complete fix

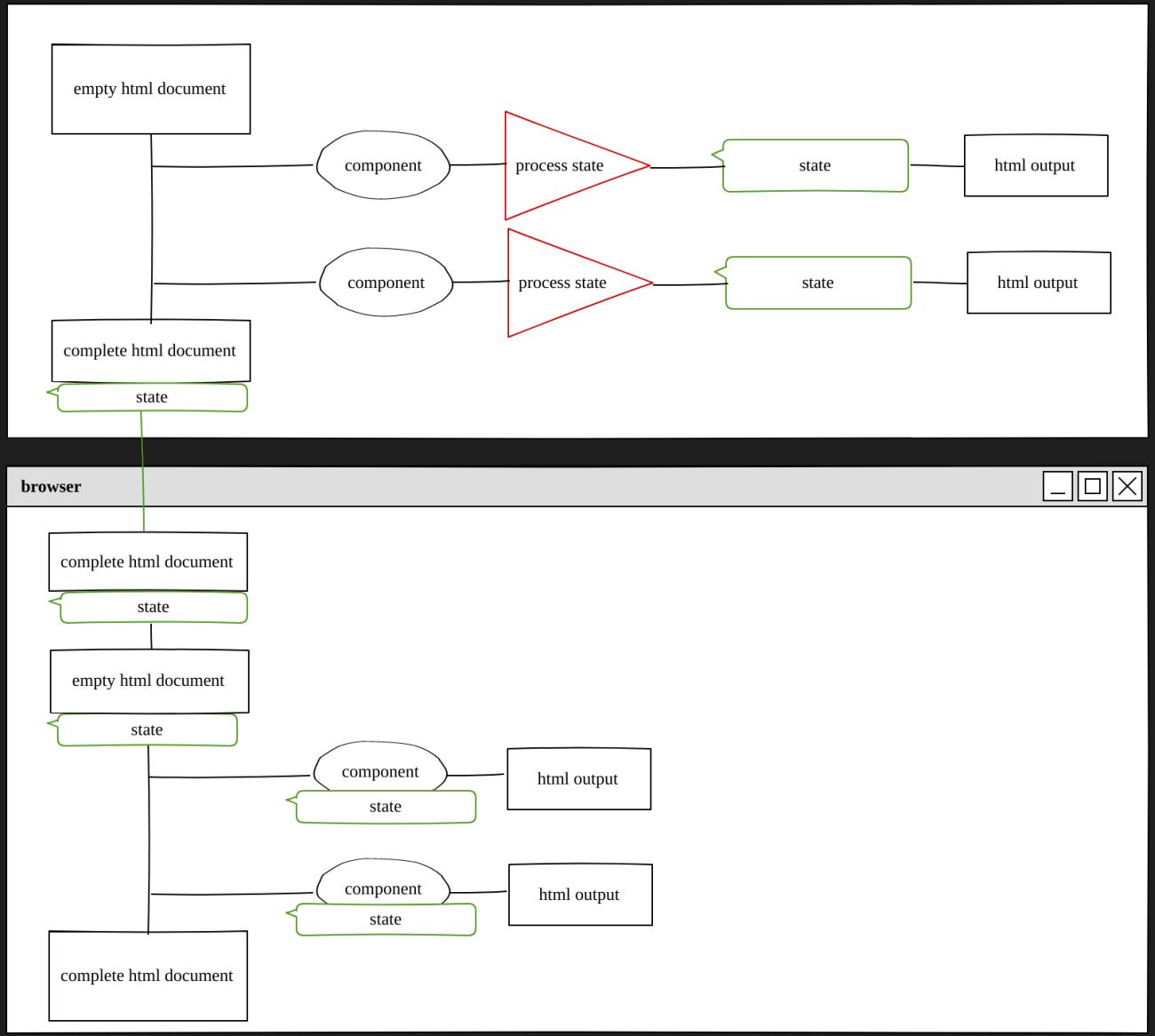
- review of other slow screens
- **make developers question 'does this need to run on the server?**'



server state transfer

- entire page render flow is duplicated - everything
- no state transfer from server to browser
- key feature of Angular SSR missing


empty html document
component
process state
state
html output
component
process state
state
html output
complete html document
browser
complete html document
empty html document
component
process state
state
html output
component
process state
state
html output
complete html document


empty html document
component
process state
state
html output
component
process state
state
html output
complete html document
state
browser
complete html document
state
empty html document
state
component
state
html output
component
state
html output
complete html document

LCP gain

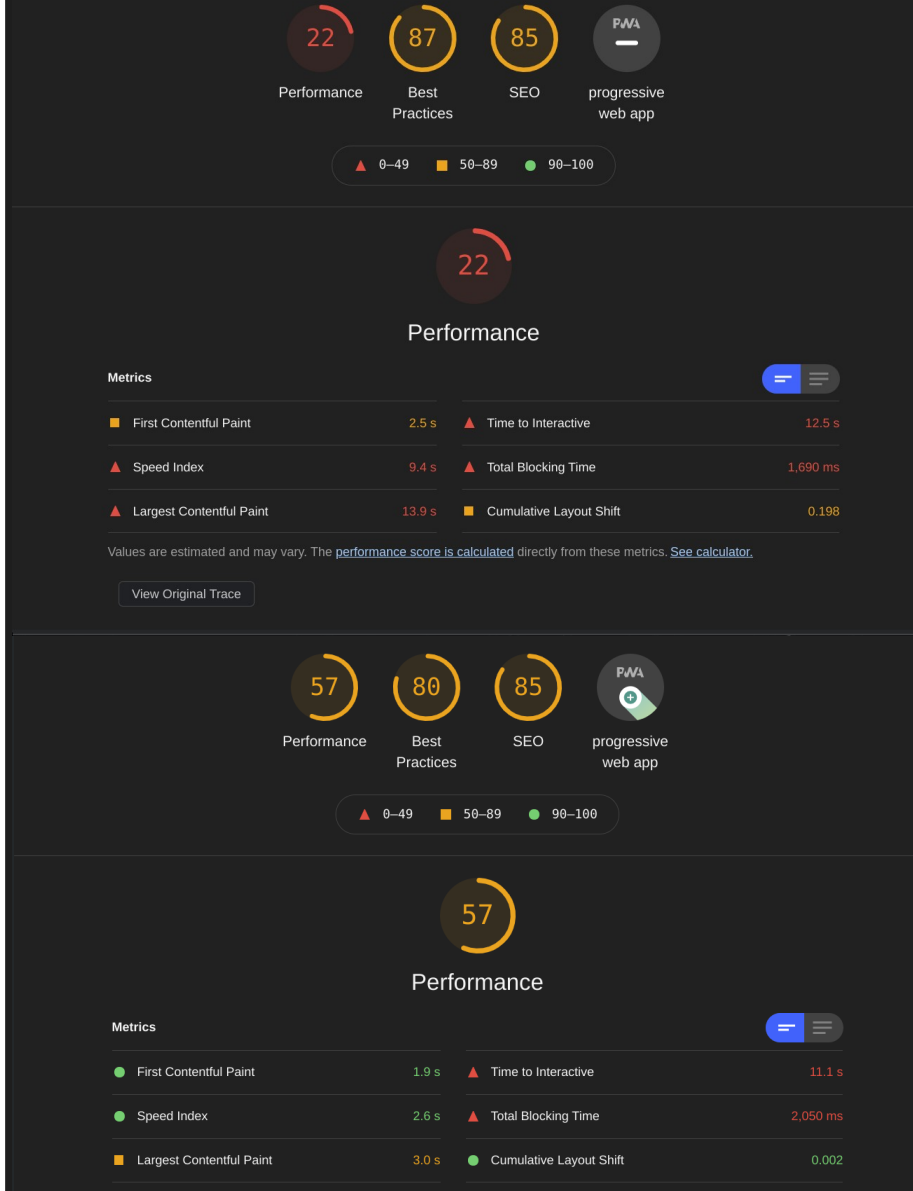
- testing being done using same scoring system as google

- in testing /resultat LCP dropped from 14s to 3s

- LCP gain of 11 seconds for average mobile devices

- biggest gain due to state transfer

- no need to create search results twice



LCP gain

- testing being done using same scoring system as google

- in testing / (homepage) LCP dropped from 9s to 3.8s
- LCP gain of 5 seconds for average mobile devices

- biggest gain due lower core JS bundle size
- non blocking render