Dokumentation

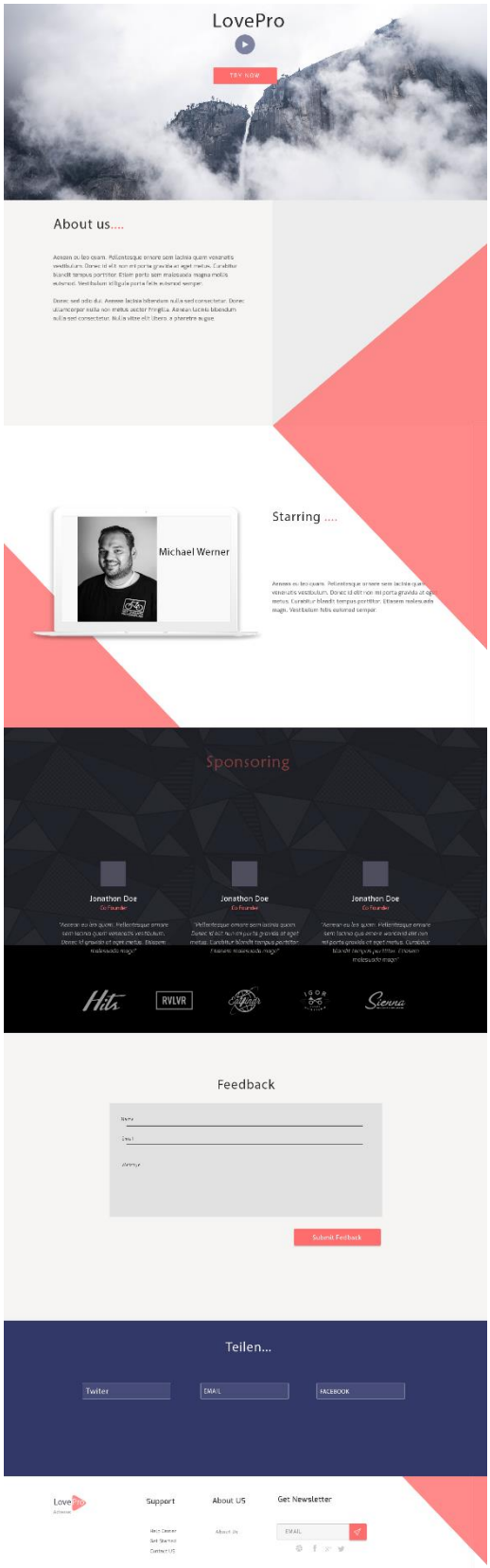
Interaktiver Webvideoplayer

Diana Agheeva MI7, 8. Semester, Mtrnr: 32567
Leon Kiefer MI7, 6. Semester, Mtrnr: 34686
Angela Strähle MI7, 6. Semester, Mtrnr: 34515

Inhaltsverzeichnis

1. Frontend

1.1 Mockups



Zu Beginn des Projektes haben wir uns mit dem StuPro Team zusammengesetzt, um herauszufinden, welche Vorstellungen diese bezüglich der Website haben. Es gab keine Vorschläge, also haben wir im Frontend unabhängig voneinander zwei verschiedene Designvorstellungen gemacht, um diese dem Team vorzustellen. Das Mockup wurde im Stil des Minimalismus erstellt. Die Kombination aller Farben wurde unter Berücksichtigung der Präsentation ausgewählt, die das Team uns gesendet hat. Da alle benötigten Elemente auf einer Seite platziert werden konnte, haben wir uns für eine einseitige Seite entschieden, sogenannte Landing Page. Landing Page - ist ein derzeit sehr beliebtes Website-Format, das keine verzweigte Struktur impliziert. Der gesamte Inhalt einer solchen Site passt auf eine einzelne Seite und wird durch Scrollen nach oben und unten angezeigt. Die Vorteile einer solchen Site sind offensichtlich:

- Landing Page ist viel schneller und billiger zu implementieren.
- Einfache Möglichkeit, ein Responsive web design zu erzielen.
- Der Betreiber einer solchen Site kann sicher sein, dass der Besucher die notwendigen Informationen erhält und nicht von etwas anderem abgelenkt wird.
- Den Eindruck der Vollständigkeit der bereitgestellten Informationen

Das Mockup besteht aus 5 Sections: Video, About us, Starring, Sponsoring

und Feedback. Die Website enthält keine große Menge an Bildmaterial, gleichzeitig werden Farben und Elemente so ausgewählt, dass die Aufmerksamkeit des Benutzers auf sich gezogen und gehalten wird.

Am oberen Rand der Seite befindet sich ein Video. Nachfolgend findet man alle weiteren Informationen. Bei der Gestaltung wurden wichtige Elemente wie Raum, Hilfslinien, Formen und Farben berücksichtigt. Die Verwendung eines vorbereiteten Rasters für das Layout ist eines der Grundprinzipien des Webdesigns, das dazu beiträgt, die gesamte Arbeit sofort auszugleichen und zu organisieren. Unser Design basierte auf einem einfachen Layout in 12 Spalten mit einer Breite von jeweils 24 Pixeln - einschließlich Absätze, die Gesamtbreite beträgt 970 Pixel. Ein nützliches Werkzeug zum Erstellen eines solchen Entwurfsrasters ist der "Grid System Generator".

Die Figuren (Formen) im vorgestellten Design sind in zwei Typen unterteilt: organisch und geometrisch.

Organisch - solche Figuren umfassen die Umrisse von Autos, der Sonne, Häusern, Möbeln und anderen Gegenständen, Kreaturen und Phänomenen, die im Leben auftreten. In unserem Design ist dies ein Laptop mit Fotos von Schauspielern.

Geometrisch- diese Art von Figur umfasst: Rechteck, Raute, Kreis, Dreieck, Quadrat usw. Sie dienen als Dekorations- und Funktionselemente wie: Buttons, Textfeld für Feedback, Tabs für active icons.

Video

Project

Duis sagittis sollicitudin nisl, porttitor convallis nunc vulputate tincidunt. Curabitur vestibulum, turpis sed luctus luctus, erat nisl imperdiet quam, ac finibus orci nulla ac urna. Nunc sit amet fermentum sem. Suspendisse tempus pulvinar urna, id faucibus est fermentum sit amet. Nam lacinia sed diam sed varius. Vivamus fermentum felis ac lectus molestie, ac gravida leo elementum.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eleifend mi sed sem semper suscipit. Sed dapibus, sem a ultrices ullamcorper, nulla augue condimentum nisl, vel congue lorem diam sed orci. Nulla porta gravida tristique. In vitae suscipit enim. Cras quis lacinia enim. Aliquam tincidunt elit quis interdum mollis. Donec a pellentesque purus. Curabitur ac libero vel ex efficitur pellentesque in nec ante. Proin scelerisque odio tortor, et pharetra orci tristique ut. Nulla in tortor nibh. Cras feugiat egestas nunc, a interdum lacus pellentesque eu. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae;

Duis sagittis sollicitudin nisl, porttitor convallis nunc vulputate tincidunt. Curabitur vestibulum, turpis sed luctus luctus, erat nisl imperdiet quam, ac finibus orci nulla ac urna. Nunc sit amet fermentum sem. Suspendisse tempus pulvinar urna, id faucibus est fermentum sit amet. Nam lacinia sed diam sed varius. Vivamus fermentum felis ac lectus molestie, ac gravida leo elementum. Donec quis faucibus nibh. Vivamus in iaculis sem. Proin bibendum vulputate sodales. Duis rutrum consequat odio, quis ultricies nulla. Integer tempus, lectus sed facilisis consectetur, purus quam ornare erat, sit amet efficitur ex ligula non augue.

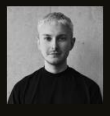
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eleifend mi sed sem semper suscipit. Sed dapibus, sem a ultrices ullamcorper, nulla augue condimentum nisl, vel congue lorem diam sed orci. Nulla porta gravida tristique. In vitae suscipit enim. Cras quis lacinia enim. Aliquam tincidunt elit quis interdum mollis. Donec a pellentesque purus. Curabitur ac libero vel ex efficitur pellentesque in nec ante. Proin scelerisque odio tortor, et pharetra orci tristique ut. Nulla in tortor nibh. Cras feugiat egestas nunc, a interdum lacus pellentesque eu. Vestibulum ante ipsum primis in faucibus orci luctus et

About us

Anna Smirnov
Regie & Drehbuch

Veit Schuele
Regie & Drehbuch

Joshua Balz
Produktionsleitung



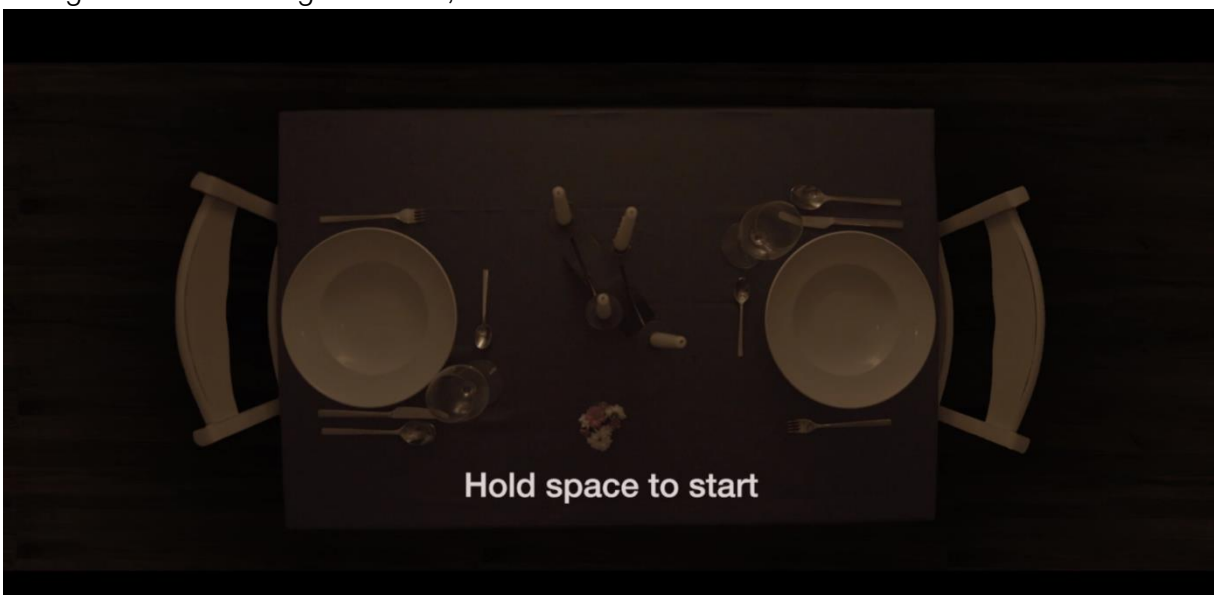
Evelin Buhman
Oberbeleuchterin

Die zweite Designvorstellung wurde anfangs verworfen, da die erste dem Team farblich besser und benutzerfreundlicher gefunden hat. Nach weiteren Wochen und fortlaufendem Prozess haben wir die Mitteilung bekommen, dass die zweite Designidee verwirklicht werden soll. Somit haben wir die zweite Projektidee überarbeitet und angepasst. Diese war die Grundidee für die jetzige Website.

Die Inspiration kam von der Website andeinerseite.video. Diese Website war auch die Inspiration für das StuPro Team einen interaktiven Webvideoplayer programmieren zu lassen. Wir wollten ähnliche Elemente, die wir auf dieser Website gut fanden, aber wollten ebenso bessere Elemente, die uns von dieser abheben lassen. Das Team unterstützte die Idee eines OnePagers. Ebenso haben Sie sich für einen Slider entschieden, um die Seite dynamisch zu halten, unabhängig vom Video. Die Elemente auf der Seite wurden alle vom StuPro Team gewählt. Anordnung, Schriftgröße, Bildgröße sowie Position der Bilder wurden mit dem StuPro Team in mehreren Terminen mit uns ausgearbeitet. Die Schriftart wurde vom Team gestellt.

1.2 Anforderungen

Das Design der jetzigen Website wurde vom Team fast wöchentlich überarbeitet. Während anfangs sehr wenig Input dazu kam, wurde es bei steigender Wochenzahl stetig erhöht. Kommunikation war hier sehr wichtig. So wurde im oberen Bereich keine Navigationsbar mehr gewünscht, sondern nur das interaktive Video.



Beim Scrollen der Website sollte eine Slidebar sichtbar sein, die jeweils nach zehn Sekunden die Inhalte wechselt. Es wurde eine Sektion der Projektbeschreibung, der Personen, die an der Produktion beteiligt waren, die Schauspieler und das Sponsoring erstellt. Außerdem wurde einen Indikator gewünscht, um die Möglichkeit zu haben, visuell zu sehen, auf welcher Sliderseite man sich aktuell befindet.

CREW

DIRECTOR	ANNA SMIRNOV VEIT SCHUELE
PRODUCER	JOSHUA BALZ
SET MANAGER	PHILIP EBERLING
GAFFER	EVELIN BUHMANN
DIGITAL IMAGING TECHNICIAN	JOSHUA BALZ
SOUND DESIGN	DAVID HETZ
COLOURIST	EVELIN BUHMANN ANNA SMIRNOV
VFX & RETOUCHING	MAXIMILIAN ECKMANN VEIT SCHUELE
WEB IMPLEMENTATION	ANGELA STRAEHLE LEON KIEFER DIANA AGHEEVA
SCREENWRITER	ANNA SMIRNOV VEIT SCHUELE
DIRECTOR OF PHOTOGRAPHY	MAXIMILIAN ECKMANN
CAMERA ASSISTANT	PATRICK BAITINGER
BEST BOY	JONATHAN LORENZ
MAKE UP ARTIST	LISA SCHUBERT
STILL PHOTOGRAPHER	JOSHUA BALZ
EDITOR	MAXIMILIAN ECKMANN
BOOM OPERATOR	FLORIAN HEIDECKER DAVID HETZ
ORIGINAL SCORE	MAJA MERZ DAVID HETZ

CAST

Christos Raptis

Marion Henkelmann

Benedikt Haefner

SPONSORING

wir für euch
Freunde & Förderer
der HdM e.V.

HOCHSCHULE
DER MEDIEN

HCAM

S'Kopierlädle

ETZOLD
STIFTUNG
WISSEN VERMITTELN

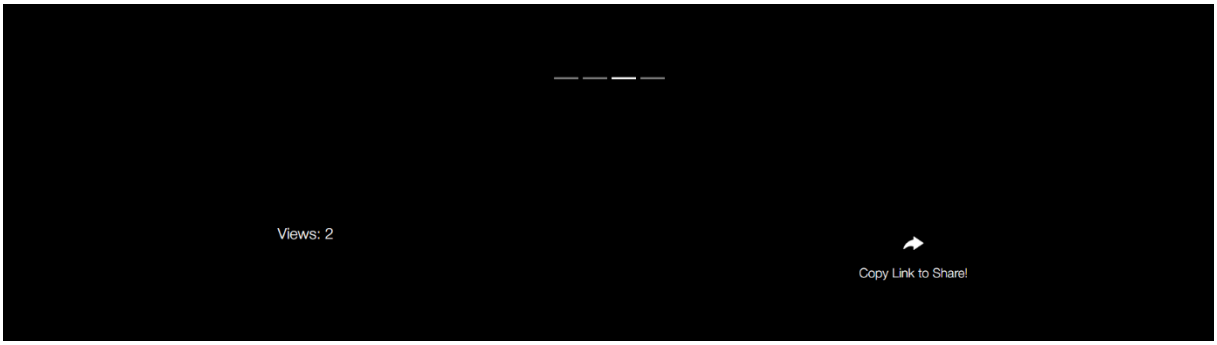
coffreez
FROZEN COFFEEBAR

eat talk

ZOLLER
DAS BACKHAUS

NEUE
DYNASTIE

Im Footerbereich soll eine Zahl stehen, die die Aufrufe der Seite zählt. Dazu soll es eine Möglichkeit geben, die URL direkt zu kopieren, um den Link teilen zu können.



1.3 Der Aufbau

Der Aufbau der Seite wurde mit HTML5 und CSS in Brackets geschrieben.

Teile, wie den Slider, wurde mit Bootstrap, sowie JavaScript verwirklicht.

Wir haben die einzelnen Sliderseiten in Sektionen unterteilt, um die Übersichtlichkeit zu bewahren. Außerdem haben wir diese in gleichgroßer Höhe und Breite gesetzt, damit der Slider beim weiter oder zurücksliden immer gleichgroß ist und somit gleich aussieht.

Den Aufbau haben wir immer in Reihen (row) und Zellen (column) angeordnet, um ein besseres Layout zu erstellen und die Positionierung zu erhalten. Die ganze Struktur wurde dadurch mit Bootstrap angeordnet.

Die Überschrift haben wir in jeder Sektion in eine eigene Reihe gepackt. Gewünscht war, dass alle Überschiften genau gleich groß und die gleiche Position haben.

In der Sektion „Projekt“ haben wir eine Reihe mit zwei Spalten. In der ersten Spalte befindet sich Platz für den Text und auf der anderen Seite eine Collage aus drei Bildern, die wir als ein einziges Bild eingefügt haben.

Bei der Bezeichnung mit den jeweiligen Personen haben wir in der Sektion „Crew“ jeweils zwei Spalten verwendet und diese in zwei Spalten gepackt. Somit hatten wir neun Reihen und vier Spalten. Die Bezeichnung hebt sich von den Personen in fetter Schrift ab. Die Anordnung musste teilweise geändert werden, da eine Reihe größer wurde, wenn mehr Personen in der Zelle standen. So musste neu angeordnet werden, damit die Sektion nicht größer wurde als die anderen Sektionen.

In der dritten Sektion haben wir in der ersten Reihe drei Bilder verwendet und diese in der nächsten Reihe mit drei Untertiteln versehen. In der letzten Sektion wurde drei Reihen auf drei Spalten verwendet, in der wir verschiedengroße Bilder eingefügt haben.

Die ganze Anordnung der Crewbezeichnung, der Größe der Bilder sowie der Position wurde genaustens mit dem StuPro Team abgestimmt.

Der Seitenzähler haben wir mit einem GET-Request mit Ajax erstellt. Dabei wird bei jedem aktualisieren der Seite einen Timestamp erzeugt, der vom Server gespeichert wird. Jeder Timestamp wird vom Server gezählt und erhöht den Gesamtzähler, der auf der Seite wieder ausgegeben wird. Auf der Adminseite wird der Timestamp, sowie die Anzahl ausgegeben. Der Timestamp wurde vom StuPro Team gewünscht.

```
function getViewCount(){

    $.ajax({
                    type: 'GET',
                    url: 'http://localhost:3000/admin/getViewCount',
                    data: { },
                    async: true,
                    success:function(data){
                        document.getElementById('resultViewCount').innerHTML = "Anzahl der Views: " + data;
                    },
                    //document.open();document.write(data);document.close();        Um neue Seite zu laden, html tags im template
                    //document.getElementsByTagName('body')[0].innerHTML = data, um neuen body anzuzeigen
                    statusCode: {
                        }
                });}
```

```
function logView(){

 //var date = Date.now().toString();
        $.ajax({
            type: 'GET',
            url: 'http://localhost:3000/admin/viewCount',
            data: {},
            async: true,
            success:function(data){

                /*document.getElementById('iframe').innerHTML = data*/
            },
            //document.open();document.write(data);document.close();        Um neue Seite zu laden, html tags im template
            //document.getElementsByTagName('body')[0].innerHTML = data, um neuen body anzuzeigen
            statusCode: {
            404: function(){
            //alert('page not found');
            }
                }
        });}
```

Serverseite:

```
app.get('/admin/viewCount',function(request, response) {

  var date = Date.now();
  //var date_format_str = date.getFullYear().toString()+"-"+((d.getMonth()+1).toS
  var date_format_str = dateFormat(date, "dddd, mmmm dS, yyyy, h:MM:ss TT");

  views.push(date_format_str);
  viewCount ++;
  console.log(views.toString() + ", ViewCount: " + viewCount);


});

app.get('/admin/GetViews',function(request, response) {
  var resultHtml = "";
  resultHtml += "<table>";
                    resultHtml += "<tr><th>" + "Views" + "</th></tr>";

                    for (var i = 0; i < views.length; i++) {
                    resultHtml += "<tr><td>" + views[i] + "</td></tr>";
                    }
                    resultHtml += "</table>";



  response.send(resultHtml);


});
```

Aktuell zählt der Zähler nur solange, wie der Server aktiv ist. Wenn der Server geschlossen wird, bricht der Zähler ab und fängt wieder bei null an. Die Funktion wurde für die Media Night gewünscht.

Außerdem war ein Button gewünscht, der den Link der Seite kopiert. Dazu haben wir einen Code in JavaScript geschrieben, um den link iloveyoutwo.com beim Klicken zu kopieren. In Zukunft wird die Seite diese Domaine tragen, so wurde es gewünscht. Dazu soll beim Klicken des Buttons ein Tooltip erscheinen, dass das Kopieren erfolgreich stattgefunden hatte.

1.4 Responsive

Da sich der Hauptinhalt unserer Seite im Slider-Bereich befindet, ist es kaum möglich die Seite für alle Bildschirauflösungen Responsive zu machen.

Problem:

Responsive Design unterscheidet sich von anderen darin, dass es sich je nach Bildschirmbreite des Geräts (Browser) anpassen (ändern) kann. Die Hauptblöcke werden auf die Breite des Bildschirms des Mobilgeräts komprimiert, wo dies nicht möglich ist - die werden zu einem langen Band zusammengefügt.

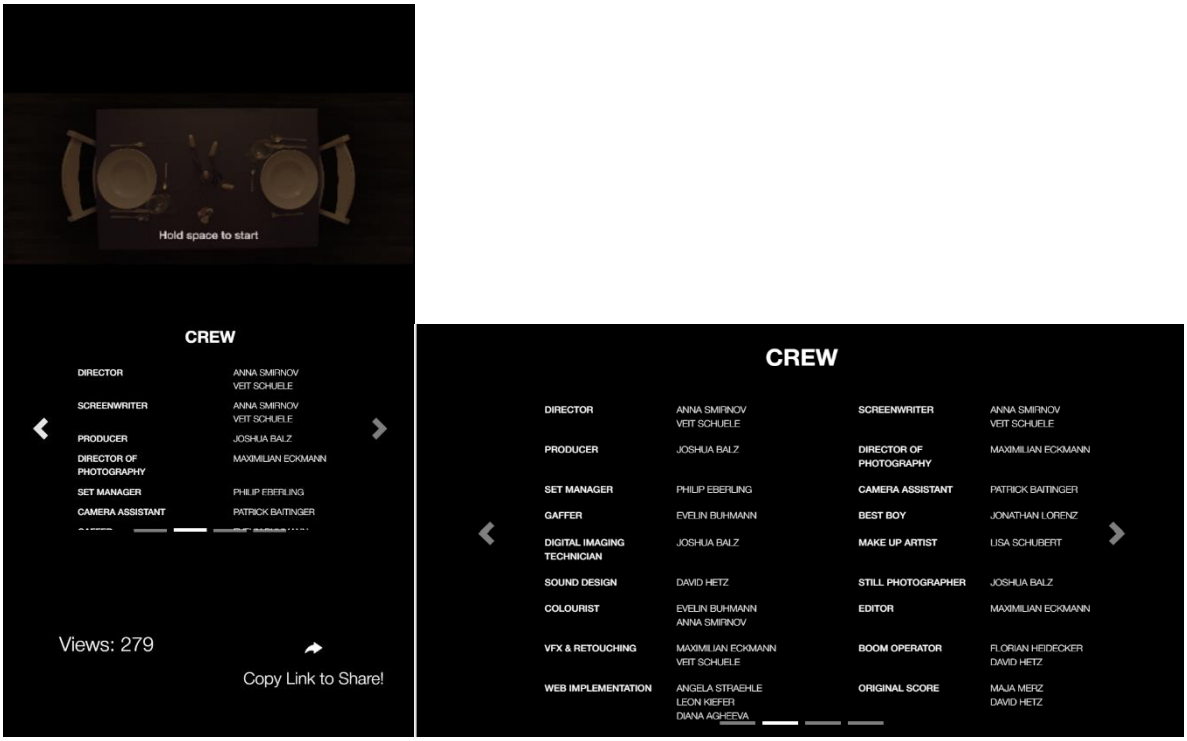
Der gewünschte Slider mit den Inhalten lässt sich nur quer betrachten, sonst werden vor allem die Schriften für den Leser zu klein. Durch die Aufstellung der Sektion Crew würde der Slider in der Mobile Ansicht sich lang nach unten bewegen, was aber vom Team nicht gewünscht wurde.

Lösung:

Wir haben die Seite responsive für die häufigsten Mobile-Bildschirmauflösungen gemacht, sowie für Computer mit weitem Monitor.

Responsive Webdesign wurde für folgende Auflösungen realisiert:

- 360 x 640
- 375 x 667 (iPhone 6/7/8)
- 1929 x 1080
- 1366 x 768



Auf Mobilgeräten wird die Site am besten im Querformat angezeigt. Da es ein offensichtliches Problem im Hochformat in Section „Crew“ gibt – aufgrund der Tatsache, dass sich die Größe des Sliders nicht an den Inhalt anpasst, sieht der Benutzer nicht alle Informationen.

Implementierung:

Um ein responsives Design zu erstellen, haben wir media queries, Bootstrap 4 und Bootstrap 3 verwendet.

Media queries - sind CSS-Regeln, mit denen die Stile von Elementen in Abhängigkeit von den Werten der technischen Parameter von Geräten gesteuert werden können. Mit anderen Worten, dies sind Konstruktionen, mit denen man unter bestimmten Bedingungen bestimmen kann, welche Stile auf der Webseite verwendet werden sollen und welche nicht.

Die folgende Syntax wird zum Erstellen von Medienabfragen verwendet:

@media condition {

/ * Stile (werden ausgeführt, wenn das Gerät die angegebene Bedingung erfüllt)

}

Gerätetypen:

- all - alle Geräte (Standard)
- print - Drucker und Seitenvorschaumodus vor dem Drucken
- screen - Geräte mit Displays

Logische Operatoren:

- and - setzt die zwingende Erfüllung aller genannten Voraussetzungen voraus.

- , (Komma) - setzt die obligatorische Erfüllung mindestens einer der in der Medienanfrage angegebenen Bedingungen voraus.
- not - soll die angegebene Bedingung verweigern. Es hat eine niedrigere Priorität in Bezug auf den Operator und, d.h. Der Operator not wird immer nach dem Operator and ausgeführt.

Medienfunktionen:

- width - gibt die Anforderungen für die Breite des Anzeigebereichs des Geräts an (Browser).
- min-width - legt die minimale Breite des Darstellungsbereichs in px, em oder anderen Einheiten fest.
- max-width - Gibt an, wie groß der maximale Arbeitsbereich des Geräts sein soll (Browser).
- Orientation – Funktion, die prüft, in welchem Modus (Hochformat oder Querformat) die Seite angezeigt wird.

```
@media screen and (min-width:400px) and (max-width: 1300px) and (orientation: portrait )
```

Die oben abgebildete Zeile zeigt, dass CSS-Stile nur dann angewendet werden, wenn die Seite auf einem Gerät mit einem Bildschirm angezeigt wird, einen Anzeigebereich von 400 und nicht mehr als 1300 Pixel aufweist und sich auch im Hochformat befindet.

Um die Seite Responsiv zu machen, haben wir auch Bootstrap benutzt. Viele Themen für Content-Management-Systeme (CMS) werden mit Bootstrap entwickelt, was sein Qualitätsniveau beweist. Tatsächlich handelt es sich um ein Framework, das Komponenten von CSS, HTML und JavaScript sowie eigene Stile und Schriftarten enthält.

Das Rastersystem von Bootstrap bietet eine schnelle und einfache Möglichkeit, die Seite responsiv und adaptiv zu machen.

```
<footer class="main">
    <div class="container-fluid">
        <div class="row">
            <div class="col-6 footer">
```

Die gesamten Raster sollten sich in einem gemeinsamen Container befinden. Es kann ein Block mit der Klasse Container oder Container-fluid sein. Der Unterschied zwischen den Klassen besteht darin, dass der erste Container maximale feste Größen aufweist, nämlich 1170 Pixel. Das heißt, die Breite der Site wird diesen Wert nicht überschreiten.

Container-fluid - ist ein vollständiges fluid Layout, das sich immer bis zu 100% der Breite des Fensters erstreckt. In diesem Container befindet sich ein weiterer Block mit einer Klasse „row“ (eine Gitterlinie). Die Zellen und die Spalten befinden sich bereits direkt in der Reihe.

Die Spalte hat die Klasse col-x-x, wobei das erste x die Bezeichnung des Geräts und das zweite die Anzahl der Spalten von 1 bis 12 ist.

Die folgenden Klassen werden benutzt:

- Col-5 & Col-7 – für „Projekt“
- Col-3 – für „Crew“
- Col-4 – für „Cast“
- Col-4 – für „Sponsoring“
- Col-6 – für „Footer“

Zusammenfassung:

Medienabfragen - waren unser Hauptwerkzeug, um die Website an mobile Geräte anzupassen. Die Funktionen von Medienabfragen beschränken sich nicht nur auf die Identifizierung mobiler Geräte, sondern können zur Erstellung eines adaptiven Layouts verwendet werden. Ein solches Layout passt sich der Auflösung des Monitors und des Browserfensters an und ändert bei Bedarf die Layoutbreite, Spaltenanzahl, Bild- und Textgröße.

Im Allgemeinen erfordert das Erstellen eigener Medienabfragen mehr Zeit und Mühe, als die Verwendung vorgefertigter Lösungen von Bootstrap.

Bootstrap verwendet Medienabfragen in großem Umfang, um Raster zu erstellen. Ein Raster in Bootstrap ist ein Standardraster mit 12 Spalten, mit dem Layouts erstellt werden, die verschiedene Bildschirmgrößen mit einfacher Syntax unterstützen. Die Arbeit mit dem Bootstrap hat jedoch einige Nachteile:

1. Die von Bootstrap angebotenen Vorlagen enthalten viel mehr Informationen, als man wirklich benötigt. Die Vorlage muss viele Optionen berücksichtigen, d.h. es muss universell sein. Wir haben dieses Problem gelöst, indem wir die Komponenten in die CSS-Datei übertragen und dort steuern.

2. Der zweite Nachteil ist das Template-Design. Mit dem Framework erstellte Websites ähneln sich häufig: Die gleiche Struktur, Schaltflächen und Navigationselemente. Wenn wir die Verwendung von Standard-Bootstrap-Lösungen minimieren, kann dieses Problem ebenfalls vermieden werden.

Die beste Lösung für die Erstellung einer guten Nutzungserfahrung (UX) auf jedem Gerät besteht darin, die eigenen Medienabfragen mit gängigen Frameworks zu kombinieren (Bootstrap, Foundation).

2. Backend

Als Backend zum Ausliefern der HTML Seite wurde NodeJS gewählt. NodeJS bietet eine gute Performance und übersichtliche Arbeitsweise, da der Server innerhalb einer Datei konfiguriert wird. NodeJS bietet ebenfalls eine serverinterne Verarbeitung von dynamischen Inhalten, da die Skripsprache JavaScript genutzt wird. Dies war im Hinblick auf die Admin- und Login-Seite sehr vorteilhaft, da die hierfür bereitgestellten Funktionen direkt im "Server-File" integriert werden konnten. Der NodeJS Server wurde mit dem Express-Framework erstellt, welches eine komfortable und schnelle Konfiguration des Servers bereitstellt. Beispielsweise kann die Auslieferung von statischen Inhalten mit Express durch Angabe des Verzeichnisses realisiert werden.

```
app.use(express.static(__dirname + '/public'));
```

Mithilfe der use()- und static()-Funktion können statische Inhalte, in diesem Fall im public-Verzeichnis, über ihren Dateinamen ausgeliefert werden. Die __dirname-Variable steht für das Verzeichnis, in dem sich die Server-Datei befindet. Auf diese Weise werden ebenfalls, die sich im ressources-Verzeichnis befindenden Videos ausgeliefert.

Requests können mit Express folgendermaßen verarbeitet werden:

```
app.get('/admin/getViewCount',function(request, response) {

  response.send(viewCount.toString());

});
```

GET-Requests die auf den URL-Suffix "/admin/getViewCount" enden, werden mit einer anonymen Funktion verarbeitet. In diesem Fall wird der ViewCount (Seitenaufrufe) mit der Response zurück an den Client geschickt.

Das Ausliefern der HTML-Seite beim Aufruf der Domain, geschieht ebenfalls mithilfe des Express Request-Handlers.

```
app.get('/', function(request, response) {
    response.sendFile(path.join(__dirname + '/public/iloveyoutwo.html'));
});
```

Im Server implementierte Funktionen:

- Auth-Funktion: Die Authentifizierungsfunktion überprüft ob die vom Client eingegebene Username-Passwort-Kombination korrekt ist und leitet den Client, insofern die Kombination korrekt ist, auf die iloveyoutwo-Webseite weiter. Falls die Daten nicht korrekt sind, wird der Client darüber informiert. Da der gedrehte Film nächstes Jahr auf Filmfestivals eingereicht werden soll, war geplant die Seite mit einem simplen Login zu schützen. Letztendlich wurde sich jedoch wieder gegen das GoLive der Seite vor den Filmfestivals entschieden. Entsprechender

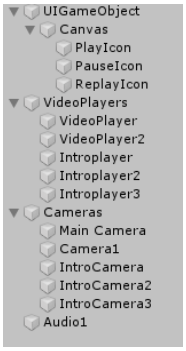
Code für das Login kann in der Serverdatei einkommentiert werden. Für den Fall, dass die Login-Seite verwendet wird, wäre der Server auf HTTPS umgestellt worden, wofür allerdings ein entsprechendes HTTPS-Zertifikat gekauft werden müsste. Für die hierfür benötigten Redirects wurde auf der von der HdM bereitgestellten virtuellen Maschine bereits ein Nginx-Server eingerichtet, welcher durch seine Geschwindigkeit und effiziente Hardwarenutzung überzeugt hat

- Erfassen des View-Counts: Um zu erfassen, wie oft und wann die Seite an der Medianight angeschaut wird, wurde ein Zähler implementiert. Der View-Count wurde mit einem GET-Request realisiert, der beim Laden der Webseite automatisch vom Client abgeschickt wird. Serverseitig wird ein Zähler hochgezählt und zusätzlich der Timestamp zur Ankunftszeit des Requests erfasst. Um den View-Count auf der Admin-Seite anzuzeigen, gibt es ebenfalls Funktionen, welche View-Count + Timestamp-Liste ausliefern.

3. Videoplayer

Für den Webvideoplayer wurde zunächst ein Prototyp mit der HTML5-Videoplayer Komponente und clientseitigem JavaScript-Code getestet. Der Prototyp wurde jedoch relativ schnell wieder verworfen, da sich der HTML5-Videoplayer nicht im gewünschten Maße konfigurieren lässt. Anstelle des HTML5-Videoplayers fiel die Entscheidung auf eine Umsetzung mit Unity. Unity bietet ebenfalls eine Videoplayer-Komponente an, welche, im Gegensatz zu der HTML5-Komponente, weitaus mehr Konfigurationsmöglichkeiten bietet.

Aufbau des Videoplayers (Unity):



Der Videoplayer besteht aus einer Szene, die in der Abbildung zusehende Komponenten enthält. Der Videoplayer enthält UI-Objekte, wie das Play-, Pause- und Replay-Icon. Die Icons sind als Image-Komponente im PNG Format eingebunden. Die Videos sind als Videoplayer-Komponenten eingebunden. Geladen werden die Videos per URL, da das Unity-WebGL-Build das Abspielen von Videos nur via URL unterstützt. Videos können nicht direkt in das Build integriert werden. Für jede Videoplayer-Komponente ist eine Kamera-Komponente vorgesehen, welche das jeweilige Video rendert. Zusätzlich besitzt der Videoplayer eine AudioSource-Komponente.

Wechsel-Logik:

Die Logik, welche das Wechseln zwischen den Videos ermöglicht, ist als C#-Skript an das Videoplayers-GameObject angehängt. Das Skript implementiert das MonoBehaviour Interface. In der Start-Methode werden alle benötigten Objekte mit den GameObject-Methoden Find() und GetComponent() instanziiert.

In der Update-Methode ist die Wechsel-Logik enthalten, welche einer großen if-else-Abfrage entspricht. Das Skript enthält eine weitere Methode (keyPressedTimer()), welche zu jedem Frame-Update misst, wie lange die Leertaste gedrückt wurde und zusätzlich ein Flag setzt, ob die Leertaste im aktuellen Frame-Update gedrückt ist.


Intro1 Intro2 Intro3

Für die Navigation durch die if-else-Schleife, wird für jedes Video ein Flag gesetzt, welches in jedem Update abgefragt wird. Zunächst steht das Flag für das erste Intro-Video auf true und die restlichen Flags auf false. Sobald die Leertaste gedrückt wird, wird das Flag für das erste Intro-Video auf false und das Flag für das nächste Video (Intro3) auf true gesetzt. Um nun Intro3 zu rendern wird die Kamera, die das erste Video rendert deaktiviert und die Kamera, welche Intro3 rendert aktiviert. Sobald Intro3, gerendert wird, wird auch die Filmmusik abgespielt. Solange Intro3, abgespielt wird, wird abgefragt, ob die Leertaste noch gedrückt ist. Wenn die Leertaste nicht mehr gedrückt ist, wird ermittelt ob die Zeit, die sie gedrückt wurde (mit keyPressedTimer() ermittelt) kleiner oder größer-gleich vier Sekunden ist. Ist die Zeit kleiner als vier Sekunden, wird Intro2, gerendert. Zudem wird die Filmmusik pausiert und auf null Sekunden gesetzt. Wird während dieses Intros die Leertaste erneut gedrückt, wird Intro3 gerendert und die Filmmusik erneut abgespielt. Sobald Intro2 das Ende des Videos erreicht und die Leertaste in diesem Zeitraum nicht gedrückt wurde, wird wieder zur ersten Intro-Sequenz, mit dem Hinweis die Leertaste zu drücken, gewechselt.

Ist die Zeit, die die Leertaste in Intro3 gehalten wurde, größer-gleich vier Sekunden, werden alle Introplayer gestoppt, beide Haupt-Videoplayer gestartet und alle Intro-Flags auf false gesetzt. Zunächst wird der erste Videoplayer gerendert. Nach den Intro-Abfragen der if-else-Schleife wird abgefragt, ob die aktuelle Zeit des einen Videos über der Länge des anderen Videos liegt. In diesem Fall wird die Kamera des längeren Videos gerendert. Dies ist notwendig, da die Videos unterschiedlich lang sind und beide Videos zu einer Spur am Ende des einen Videos zusammenlaufen. Da sich diese else-if-Abfrage im Code über der else-if-Abfrage, mit welcher zwischen den Haupt-Videos gewechselt werden kann, befindet, ist garantiert, dass während des gemeinsamen Endes beider Spuren nicht mehr gewechselt werden kann. Muss nicht zwangsweise nur eine Kamera gerendert werden, erreicht die if-else-Schleife den Teil, in welchem zwischen den beiden

Haupt-Videos gewechselt werden kann. Wird aktuell das erste Video gerendert und die Leertaste gedrückt, wird die aktive Kamera deaktiviert und die Kamera des zweiten Videos gerendert. Wird das zweite Video gerendert, jedoch die Leertaste nicht mehr gedrückt, wird die Kamera des zweiten Videos deaktiviert und die Kamera des ersten Videos aktiviert. Bei jedem Wechsel wird abgefragt, ob die zeitliche Differenz beider Videos über 0,5 Sekunden liegt. Ist dies der Fall werden die Videos zeitlich synchronisiert. Diese Toleranz ist notwendig, da der Übergang der Videos sonst nicht flüssig erscheint.

UI:

Für das Ein-und Ausblenden der UI-Elemente sorgt ein an das Canvas, welches die UI Elemente enthält, angehängtes Skript. Dieses implementiert ebenfalls das MonoBehaviour-Interface. In der Update()-Methode des Skripts wird abgefragt, ob das längere Haupt-Video sein Ende erreicht hat. Ist dies der Fall, wird das Replay-Icon eingeblendet. Ebenfalls in der Update()-Methode abgefragt wird, ob auf das Canvas geklickt wurde. Wurde auf das Canvas geklickt und der Benutzer befindet sich weder in der Intro-Sequenz, noch sind die Videos nicht pausiert, werden beide Videos pausiert und das Play-Icon für eine Sekunde eingeblendet. Umgekehrt werden beide Videos abgespielt, wenn sie pausiert sind und das Pause-Icon für eine Sekunde eingeblendet. Falls das Replay-Icon aktiv ist, wird es bei jedem Klick deaktiviert. Ein Sonderfall beim Abspielen der Videos tritt ein, sobald der Teil im Video erreicht ist, ab welchem nur noch ein Video gerendert werden darf. In diesem Fall darf nur das längere der beiden Videos, abgespielt werden. Ist das längere Video ebenfalls beendet, müssen wieder beide Videos mit einem Klick auf das Canvas abgespielt werden.

Probleme:

- Audio:

  Ein Problem beim Wechseln zwischen beiden Videos war das Audio. Da beide Videos Musik enthielten, war beim Wechseln zum einen ein manchmal Knacken, als auch ein Echo zu hören. Das Echo ist vermutlich darauf zurückzuführen, dass die mute-Operation länger dauert als die unmute-Operation.

  Lösung:

  Da die Filmmusik in beiden Videos dieselbe ist und sich das Audio nur in Soundeffekten unterscheidet, war die Lösung des Problems die Trennung von Musik und Soundeffekten. Die Musik ist getrennt als AudioSource-Komponente eingebunden und wird während beiden Videos abgespielt. Beim Wechsel werden die Videos zwar immer noch gemuted, jedoch ist kein Echo mehr zu hören da die Soundeffekte zeitlich nicht auf einander liegen. Das Knacken beim Wechsel ist durch diese Lösung ebenfalls verschwunden.

- WebGL Performance und Browser-Differenzen:

  Ein weiteres Problem stellt(e) die Performance des Videoplayers als WebGL-Build dar, da der Videoplayer als WebGL-Build im Browser deutlich mehr Ressourcen beanspruchte als im Unity-Editor. Um dem entgegenzuwirken wurden die Player Settings des Builds angepasst. Logging wurde komplett deaktiviert, da Logging laut Internetrecherchen sehr viele Ressourcen in Anspruch nimmt. Des Weiteren wurden die Quality-Settings des Builds auf eine geringere Stufe gesetzt, sowie nicht benötigte Komponenten deaktiviert. Wichtig hierbei war, Antialiasing beizubehalten, da sonst unerwünschte Flimmereffekte auftreten. Es wurde ebenfalls versucht die FPS zu “throttlen” (begrenzen), wodurch der Ressourcenverbrauch des WebGL-Builds stark verringert wurde. Allerdings traten durch das Begrenzen der FPS in den meisten Browsern ungewünschte Stotter-Effekte auf (in Chrome z.B. nicht).

  Generell ist die Performance des Videoplayers in den einzelnen Browsern sehr unterschiedlich. Firefox beansprucht sehr viel GPU-Ressourcen, wohingegen Chrome viele CPU- und GPU-Ressourcen beansprucht. In Microsoft Edge läuft der Videoplayer am performantesten.

  Zur Identifizierung von Problembereichen im Code wurde der Unity-Profiler genutzt. Mithilfe des Profilers konnten Anomalien entdeckt und im Code behoben werden. Sogar im WebGL-Build mithilfe des Debug-Builds.

4. Auswertung und Nutzerfeedback

Wir haben zur MediaNight die Anzahl der Nutzer gezählt und ihr Feedback erhalten.

Insgesamt haben wir von 230 Nutzern das Feedback erhalten, welches durchgehend positiv war. Das Nutzererlebnis unterscheidet sich dadurch, ob der Nutzer bereits vorher den Film im linearen Schnitt gesehen hatte, oder nicht. Dadurch, dass es Nutzer gab, die diesen nicht gesehen hatten, haben Sie das Video teilweise ein zweites oder drittes Mal angeschaut, um „verlorene“ Szenen noch einmal anzuschauen. Wir haben das Feedback erhalten, dass es sehr viel Spaß gemacht hat oder dass es eine nette Spielerei ist. Wiederum haben wir als Feedback erhalten, dass die Nutzer sich dabei als Regisseur fühlen, die den Cut zum Scenenwechsel veranlassen. Natürlich haben wir ebenso schlechtes Feedback erhalten, die aber an sich nichts mit der Website zu tun hatte, sondern mit der Idee, wieso wir überhaupt einen interaktiven Webvideoplayer programmiert hatten. Viele waren beeindruckt darüber, dass das Video flüssig lief, obwohl diese teilweise die Leertaste ununterbrochen getippt hatten. Der Härtetest wurde von sehr, sehr vielen Personen durchgeführt.

Für uns war die Media Night ein voller Erfolg, und wir waren positiv überrascht, wie gut und gerne die Nutzer sich das Video angeschaut haben.

Anbei sind noch Ausschnitte der Adminseite zu sehen, die die Timestamps und die Gesamtanzahl der Seitenviews enthält:

PC1:

Dashboard - iloveyoutwo

Views: 138

Gesehen am:

Thursday, January 30th, 2020, 4:43:52 PM
Thursday, January 30th, 2020, 4:45:52 PM
Thursday, January 30th, 2020, 4:48:35 PM
Thursday, January 30th, 2020, 4:50:35 PM
Thursday, January 30th, 2020, 5:08:08 PM
Thursday, January 30th, 2020, 5:08:56 PM
Thursday, January 30th, 2020, 5:10:56 PM
Thursday, January 30th, 2020, 5:21:42 PM
Thursday, January 30th, 2020, 5:21:46 PM
Thursday, January 30th, 2020, 5:22:01 PM
Thursday, January 30th, 2020, 5:24:01 PM
Thursday, January 30th, 2020, 5:26:01 PM
Thursday, January 30th, 2020, 5:28:01 PM
Thursday, January 30th, 2020, 5:30:01 PM
Thursday, January 30th, 2020, 5:36:28 PM
Thursday, January 30th, 2020, 5:38:28 PM
Thursday, January 30th, 2020, 5:40:28 PM
Thursday, January 30th, 2020, 5:42:28 PM
Thursday, January 30th, 2020, 5:56:01 PM
Thursday, January 30th, 2020, 5:58:01 PM
Thursday, January 30th, 2020, 6:01:47 PM
Thursday, January 30th, 2020, 6:03:47 PM
Thursday, January 30th, 2020, 6:07:52 PM
Thursday, January 30th, 2020, 6:09:52 PM
Thursday, January 30th, 2020, 6:13:38 PM

PC2:

Dashboard - iloveyoutwo

Views: 99

Gesehen am:

Thursday, January 30th, 2020, 5:21:49 PM
Thursday, January 30th, 2020, 5:23:49 PM
Thursday, January 30th, 2020, 5:36:11 PM
Thursday, January 30th, 2020, 5:38:11 PM
Thursday, January 30th, 2020, 5:49:22 PM
Thursday, January 30th, 2020, 5:51:22 PM
Thursday, January 30th, 2020, 6:04:15 PM
Thursday, January 30th, 2020, 6:06:15 PM
Thursday, January 30th, 2020, 6:07:08 PM
Thursday, January 30th, 2020, 6:09:08 PM
Thursday, January 30th, 2020, 6:12:05 PM
Thursday, January 30th, 2020, 6:14:05 PM
Thursday, January 30th, 2020, 6:16:05 PM
Thursday, January 30th, 2020, 6:18:05 PM
Thursday, January 30th, 2020, 6:20:05 PM
Thursday, January 30th, 2020, 6:26:37 PM
Thursday, January 30th, 2020, 6:28:37 PM
Thursday, January 30th, 2020, 6:36:07 PM
Thursday, January 30th, 2020, 6:38:07 PM
Thursday, January 30th, 2020, 6:47:07 PM
Thursday, January 30th, 2020, 6:49:07 PM
Thursday, January 30th, 2020, 6:51:07 PM
Thursday, January 30th, 2020, 6:56:12 PM
Thursday, January 30th, 2020, 6:58:12 PM
Thursday, January 30th, 2020, 7:00:12 PM