

# ECHO

Manaus Transez  
3 MMP - NMD

# Web App & REST API



React.js / Redux



Node.js / Express



MongoDB / Mongoose

# Main features

Find and discuss music


Direct messaging


Create playlists

But more importantly...

# Night Mode!

[Discussion](#) [Browse](#) [Messages](#)







**manaus\_t**


I enjoy many kinds of music. My favourite genres are heavy metal and hip hop. Among my favourite albums are ...And Justice For All by Metallica and Rust In Peace by Megadeth.


21 Posts1 Comments


[edit profile](#)





**One**  
Metallica  
by  7 days ago





**Fade To Black (Remastered)**  
Metallica  
by  7 days ago




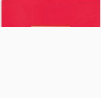
**Orion (Remastered)**  
Metallica  
by  7 days ago




**Countdown to Extinction (Deluxe Edition)**  
Megadeth  
by  7 days ago





**Beauty Behind The Madness**  
The Weeknd  
by  7 days ago




**My Beautiful Dark Twisted Fantasy**  
Kanye West

[Music](#)

[Profile](#)

[Playlists](#)


[Toggle Night Mode](#)


[Settings](#)

[Privacy Policy](#)

[Logout](#)

[Discussion](#) [Browse](#) [Messages](#)







**manaus\_t**


I enjoy many kinds of music. My favourite genres are heavy metal and hip hop. Among my favourite albums are ...And Justice For All by Metallica and Rust In Peace by Megadeth.


21 Posts1 Comments


[edit profile](#)





**One**  
Metallica  
by  7 days ago





**Fade To Black (Remastered)**  
Metallica  
by  7 days ago




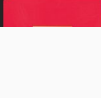
**Orion (Remastered)**  
Metallica  
by  7 days ago




**Countdown to Extinction (Deluxe Edition)**  
Megadeth  
by  7 days ago





**Beauty Behind The Madness**  
The Weeknd  
by  7 days ago




**My Beautiful Dark Twisted Fantasy**  
Kanye West

[Music](#)

[Profile](#)

[Playlists](#)

[Toggle Night Mode](#)

[Settings](#)

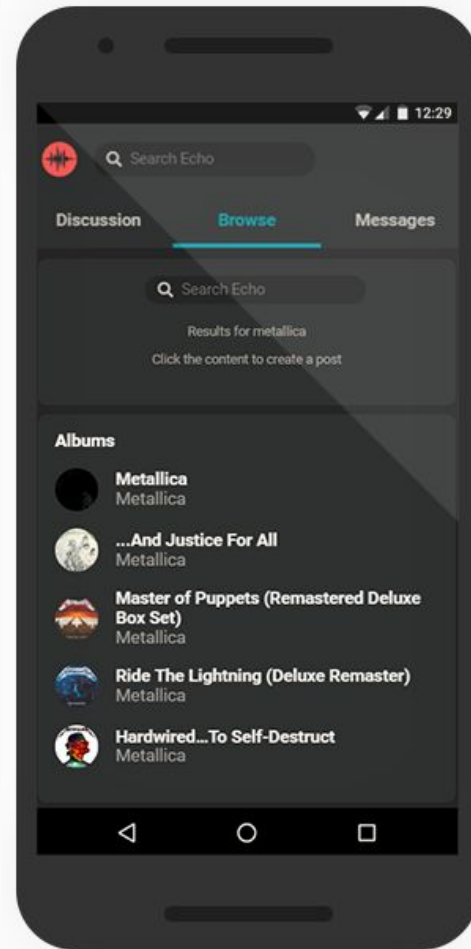
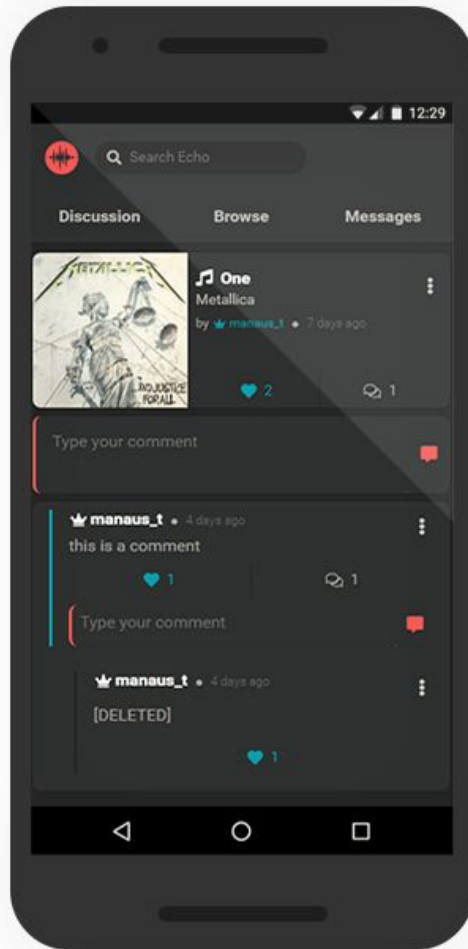
[Privacy Policy](#)

[Logout](#)

# Screenshots

Post Detail

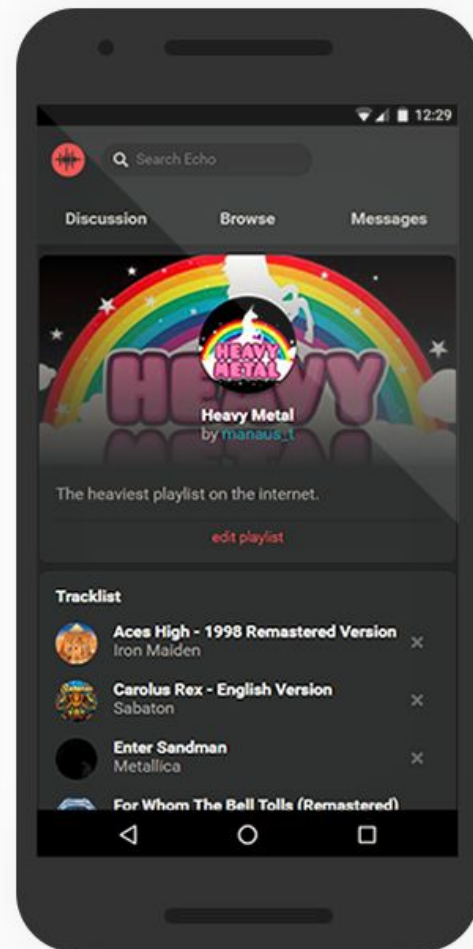
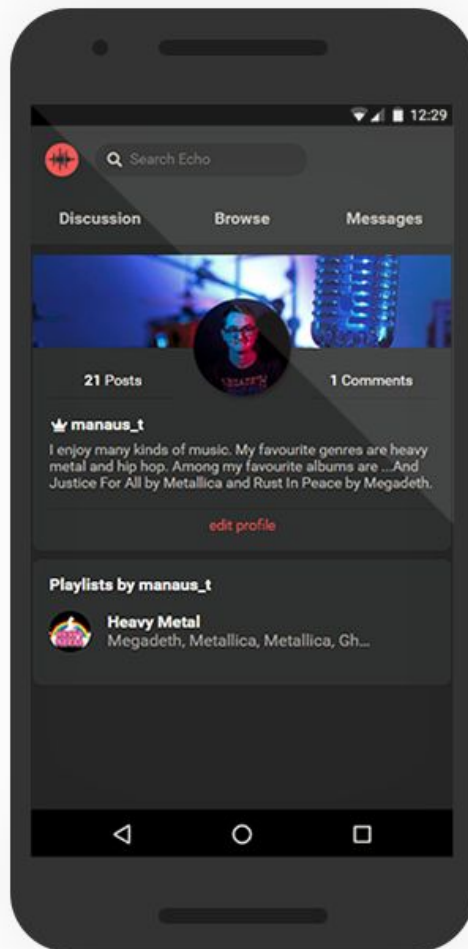
Browse Page



# Screenshots

Profile Detail

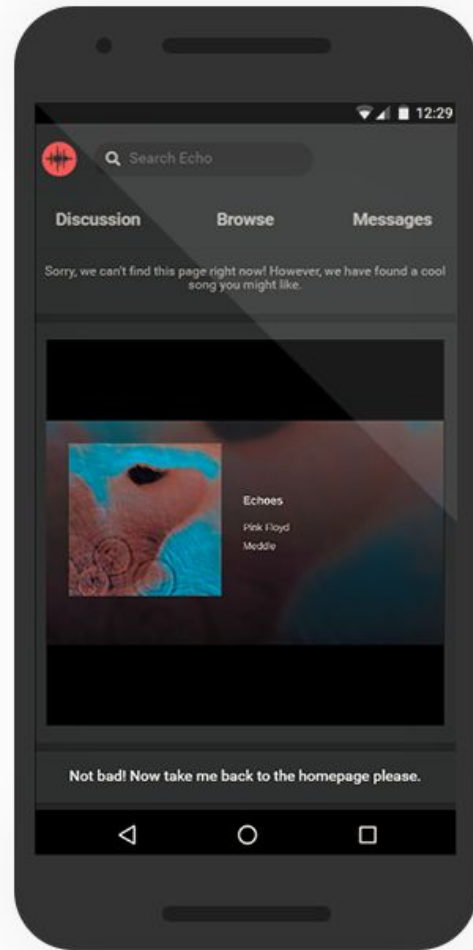
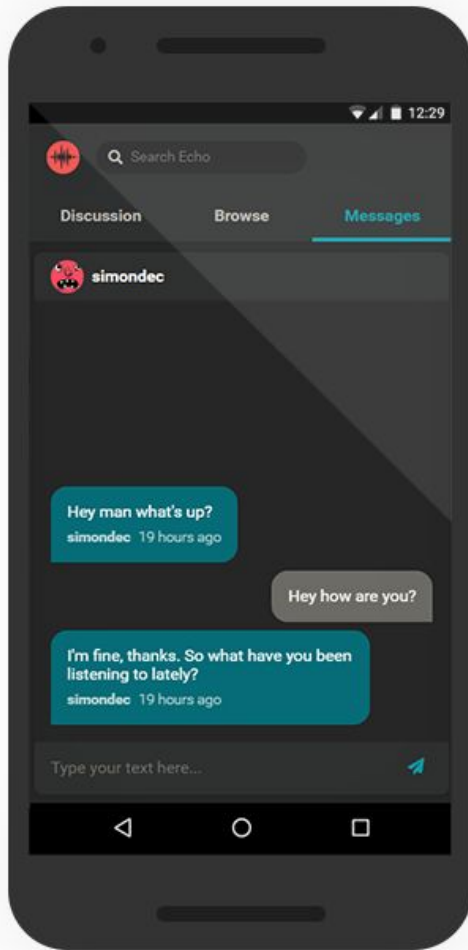
Playlist Detail



# Screenshots

Message Detail

404 Error Page





# Deliverables

# Datamodel

## 1 - 1 relation

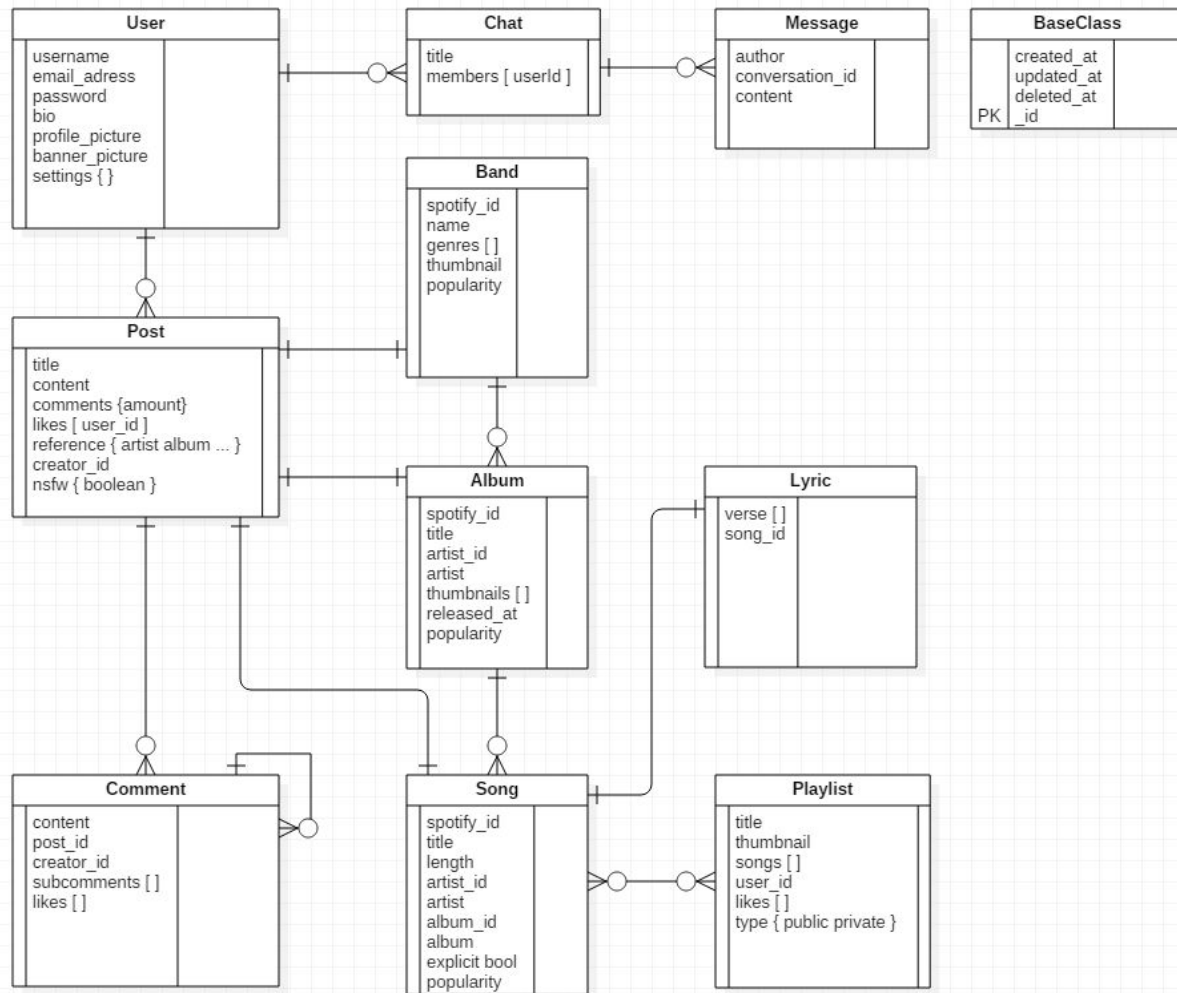
Post refers to one album, album can only have one post

## 1 - X relation

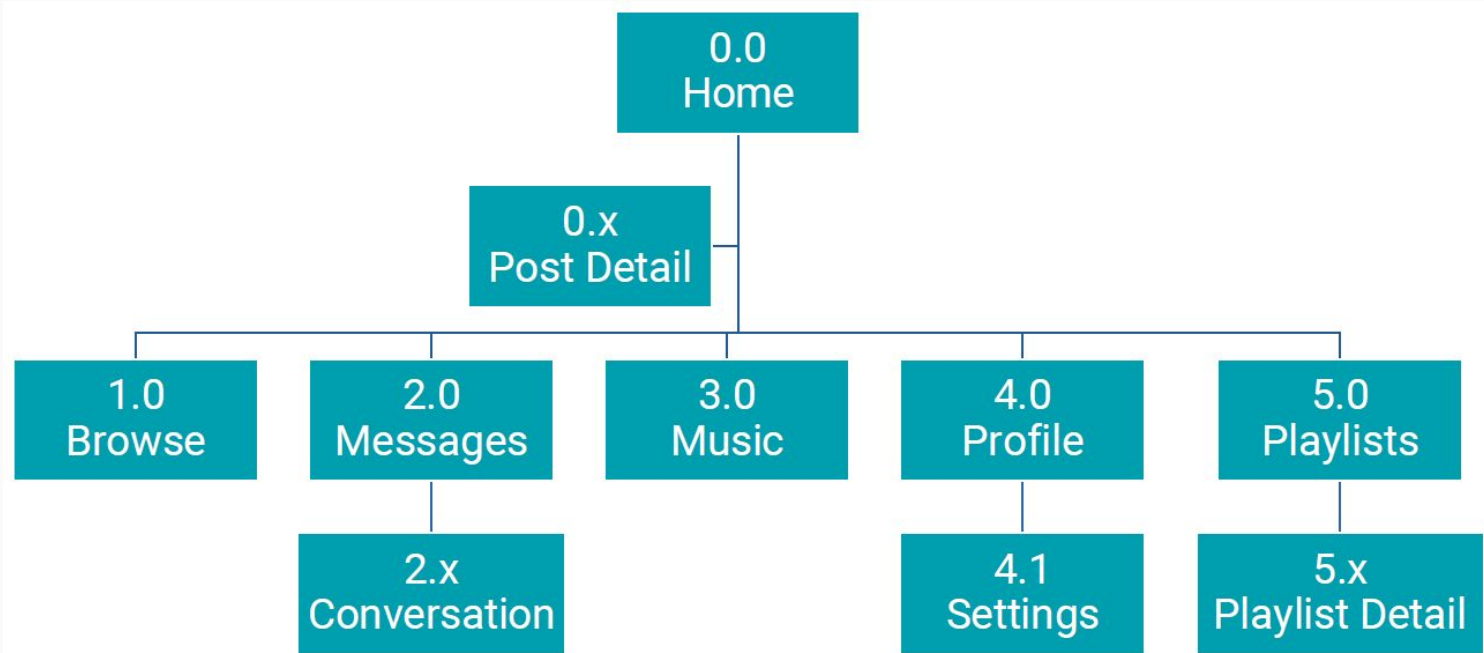
Album has many songs, song can belong to only one album

## X - X relation

Playlist has many songs, song can belong to many playlists

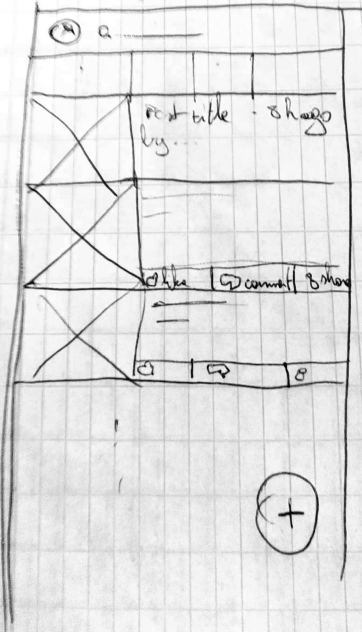


# Sitemap

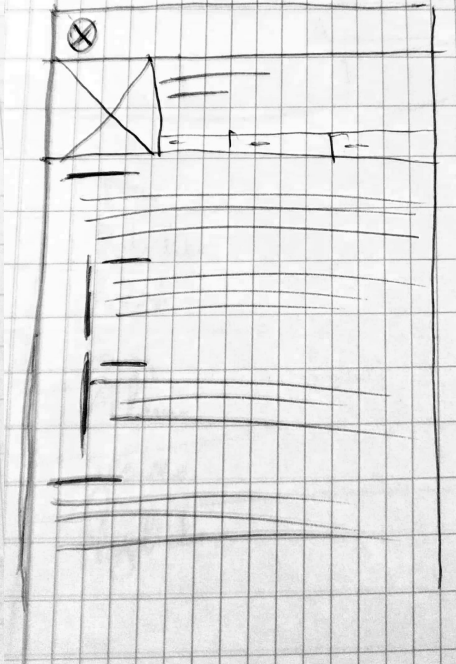


# Wireframes

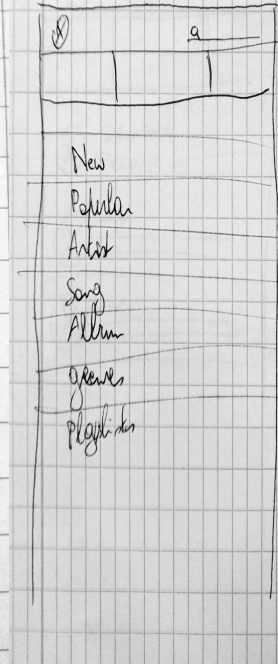
0.0 Homepage



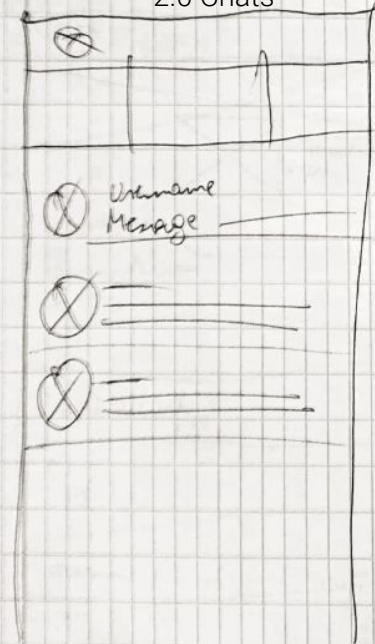
0.x Post Detail



1.0 Browse



2.0 Chats

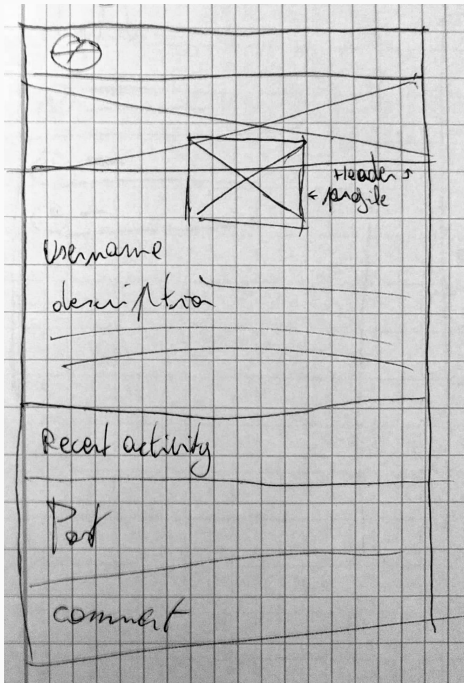


2.x Chat Detail

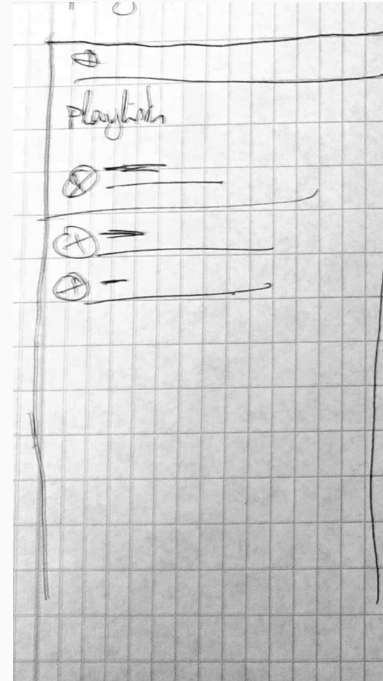


# Wireframes

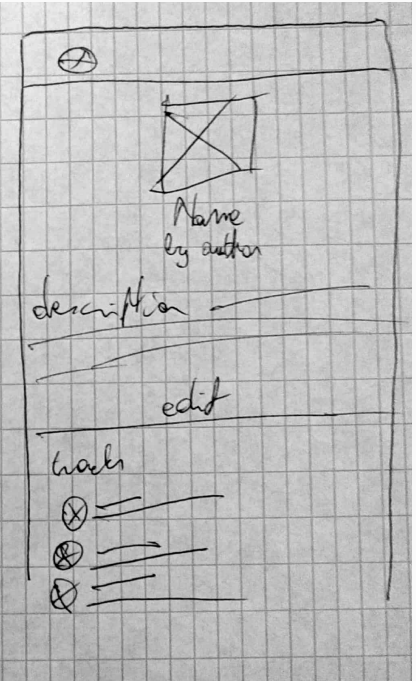
4.0 Profile



5.0 Playlists

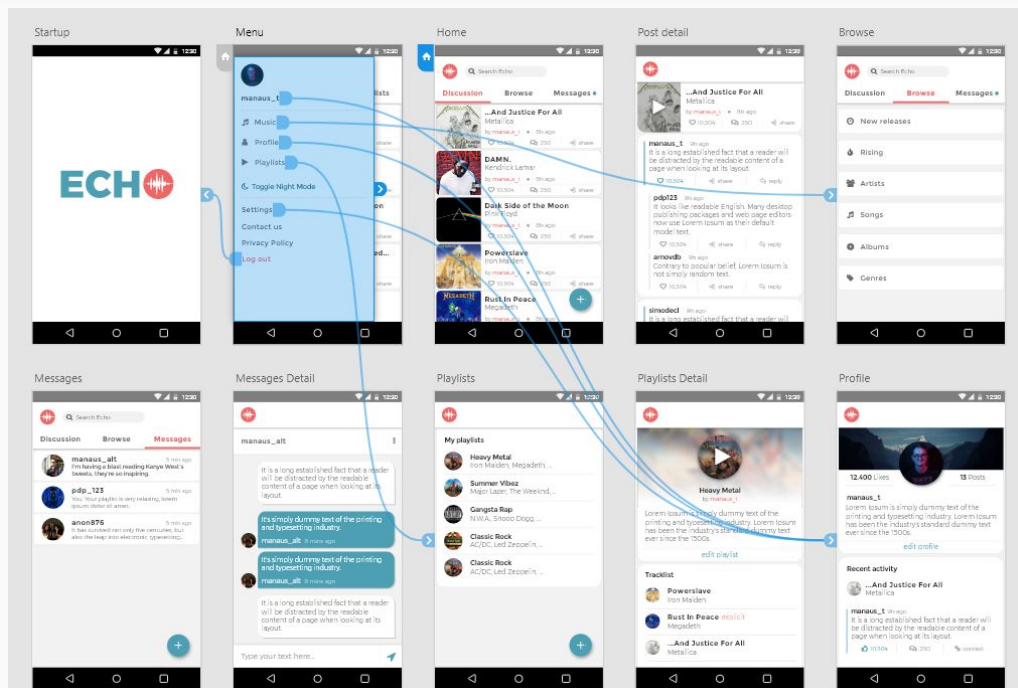


5.x Playlist Detail



# Wireflow

Wireflow preview



# Academic Poster

**Mobile Development II**  
Academic Year 2017-18  
Manaus Transez

New Media Development  
Bachelor of Graphical and Digital Media  
Artevelde University College



# ECHO

Let It Resonate

**ECHO** is a community-driven platform designed for music lovers. The application offers a hub for people who enjoy discussion and exploring their musical taste. No matter what genres you like, everyone is free to speak their mind and expand their musical horizon.

**Explore** new music, find new playlists created by other users

**Expand** your musical horizon, discover new genres and artists

**Express** your opinion and engage in a respectful discussion with like-minded people

**Export** your playlists to Spotify to enjoy anywhere

 **Node.js**  
server

 **MongoDB**  
database

 **React.js**  
client





# UI Style Guide

## Typography

Montserrat Bold  
Montserrat Regular  
Montserrat Light

## Icon



standard



alternate

## Color



primary  
#4D9FB4



secondary  
#F16667



primary alt  
#376F7F



dark grey  
#333333



grey  
#6D6D6D



light grey  
#CCCCCC

## Logo



light background



dark background

## Components



light post



dark post

## Icons

fontawesome





```
1 const mongoose = require('mongoose');
2 const Schema = mongoose.Schema;
3
4 const AlbumSchema = new Schema(
5   {
6     spotify_id: { type: String, required: true, max: 128, unique: true },
7     title: { type: String, required: true, max: 128 },
8     artist: { type: String, required: true, max: 128 },
9     artist_name: { type: String, required: true, max: 128 },
10    images: [{
11      height: { type: Number },
12      width: { type: Number },
13      url: { type: String, max: 128 }
14    }],
15    release_date: { type: String },
16    created_at: { type: Date, default: Date.now },
17    updated_at: { type: Date, default: Date.now },
18    deleted_at: { type: Date, required: false }
19  },
20  {
21    toJSON: { virtuals: true },
22    toObject: { virtuals: true }
23  }
24 );
25
26 AlbumSchema.virtual('id').get(() => this._id);
27
28 module.exports = mongoose.model('Album', AlbumSchema);
```

```
1 const mongoose = require('mongoose');
2 const Schema = mongoose.Schema;
3
4 const CommentSchema = new Schema(
5   {
6     post_id: { type: mongoose.Schema.Types.ObjectId, ref: 'Post', required: true },
7     author: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
8     content: { type: String, max: 512, required: true },
9     likes: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
10    created_at: { type: Date, default: Date.now },
11    updated_at: { type: Date, default: Date.now },
12    deleted_at: { type: Date, required: false },
13    published_at: { type: Date, required: false }
14  },
15  {
16    toJSON: { virtuals: true },
17    toObject: { virtuals: true }
18  }
19 );
20
21 CommentSchema.virtual('id').get(() => this._id);
22 CommentSchema.virtual('subcomments', {
23   ref: 'Subcomment',
24   localField: '_id',
25   foreignField: 'parent_id'
26 })
27
28 module.exports = mongoose.model('Comment', CommentSchema);
```

# Code Snippets - Controllers

postController.js  
songController.js

```
246  /*
247  Like a post
248  */
249  exports.post_like_post = function (req, res, next) {
250    User.findById(req.user.id).then(profile => {
251      console.log(req.params.postId)
252      Post.findOne({_id: req.params.postId})
253        .then(post => {
254          if (post.likes.filter(like => like.toString() === req.user.id).length > 0) {
255            // Get remove index
256            const removeIndex = post.likes
257              .map(item => item.toString())
258              .indexOf(req.user.id);
259
260            // Splice out of array
261            post.likes.splice(removeIndex, 1);
262            // Save
263            post.save().then(post => res.json(post));
264          } else {
265            // Add user id to likes array
266            post.likes.unshift(req.user.id);
267            post.save().then(post => res.json(post));
268          }
269        })
270      .catch(err => res.status(404).json({ postnotfound: 'No post found' }));
271    })
272    .catch(err => res.status(404).json({ usernotfound: 'No user found' }));
273  }
```

```
55  Get a certain song
56  */
57  exports.get_song = function (req, res, next) {
58    const id = req.params.songId;
59    const query = Song.aggregate([
60      { $match: { 'spotify_id': id } },
61      {
62        $project: {
63          "spotify_id": "$spotify_id",
64          "title": "$title",
65          "album": "$album",
66          "artist": "$artist",
67          "artist_name": "$artist_name",
68          "explicit": "$explicit",
69          "duration": "$duration",
70          "popularity": "$popularity",
71          "created_at": "$created_at",
72          "updated_at": "$updated_at",
73          "deleted_at": "$deleted_at"
74        }
75      },
76      { $lookup: { from: 'albums', localField: 'album', foreignField: 'spotify_id', as: 'album' } },
77      {
78        $project: {
79          "spotify_id": 1,
80          "title": 1,
81          "artist": 1,
82          "artist_name": 1,
83          "explicit": 1,
84          "duration": 1,
85          "popularity": 1,
86          album: { $arrayElemAt: ['$album', 0] },
87          "created_at": 1,
88          "updated_at": 1,
89          "deleted_at": 1
90        }
91      }
92    ])
93    query.exec((err, song) => {
94      if (err) return errorHandler.handleAPIError(500, 'Could not get the song with id: ${id}', next);
95      if (!song) {
96        return errorHandler.handleAPIError(404, 'Song not found with id: ${id}', next);
97      }
98      return res.json(song[0]);
99    });
100  }
```

```
import { TOGGLE_NIGHTMODE, ADD_NIGHTMODE, REMOVE_NIGHTMODE } from '../constants';

const body = document.querySelector('html');

export const toggleNightmode = () => dispatch => {
  body.classList.toggle('dark')
  let nightmode = JSON.parse(localStorage.getItem('nightmode'))
  localStorage.setItem('nightmode', JSON.stringify(!nightmode))
  return { type: TOGGLE_NIGHTMODE }
}

export const addNightmode = () => dispatch => {
  body.classList.add('dark')
  localStorage.setItem('nightmode', 'true')
  return { type: ADD_NIGHTMODE }
}

export const removeNightmode = () => dispatch => {
  body.classList.remove('dark')
  localStorage.setItem('nightmode', 'false')
  return { type: REMOVE_NIGHTMODE }
}
```

```
const initialState = {
  nightmode: false
}

function nightmodeReducer(state = initialState, action) {
  switch (action.type) {
    case TOGGLE_NIGHTMODE:
      return {
        ...state,
        nightmode: !state.nightmode
      };
    case ADD_NIGHTMODE:
      return {
        ...state,
        nightmode: true
      };
    case REMOVE_NIGHTMODE:
      return {
        ...state,
        nightmode: false
      };
    default:
      return state;
  }
}

export default nightmodeReducer;
```

```
class PostsList extends Component {
  constructor(props) {
    super(props);

    this.state = {
      comments: []
    }
  }

  componentWillMount() {
    this.props.fetchPosts();
  }

  componentDidMount() {
    let scrollTop = document.getElementById('scroll-top')
    if (scrollTop) {
      scrollTop.addEventListener('click', (e) => {
        e.preventDefault();
        window.scrollTo({ top: 0, left: 0, behavior: 'smooth' })
      })
    }
  }
}
```

```
<section className="card playlists">
  <h2>Playlists by {this.state.profile.username}</h2>
  {(this.state.playlists.length > 0)
    ? this.state.playlists.map((playlist, i) => (
      <a href={` /playlist/${playlist._id}`} key={playlist._id}>
        <div className="playlist-item" >
          <img src={playlist.image} alt="Thumbnail" />
          <div>
            <h3>{playlist.title}</h3>
            <p>
              {playlist.songs.map((song, i) => <React.Fragment key={i}>
                {!!i && ", "}
                {song.artist_name}
              </React.Fragment>)}
            </p>
          </div>
        </div>
      </a>
    ))
    : <p>No playlists yet!</p>
  }
</section>
```

# Live preview



<http://localhost:3000>

# Screencast

---

<https://vimeo.com/272941834>



# Thanks!

A presentation by:

Manaus Transez

Feel free to ask questions.

