

# <CODE>KINGS>

**Yens Broothaers**

2MMP - NMD

# **> Probleemstelling**

- **Voldoende tutorialwebsites**
- **Maar meestal betalend**
- **Redelijk duur/maandelijks betalen**

# > Oplossing

- **Community-gedreven**
- **Leden maken zelf de content**
- **Toezicht houden**

# > Technologiën



**MongoDB**  
Database



**Node.js**  
Server

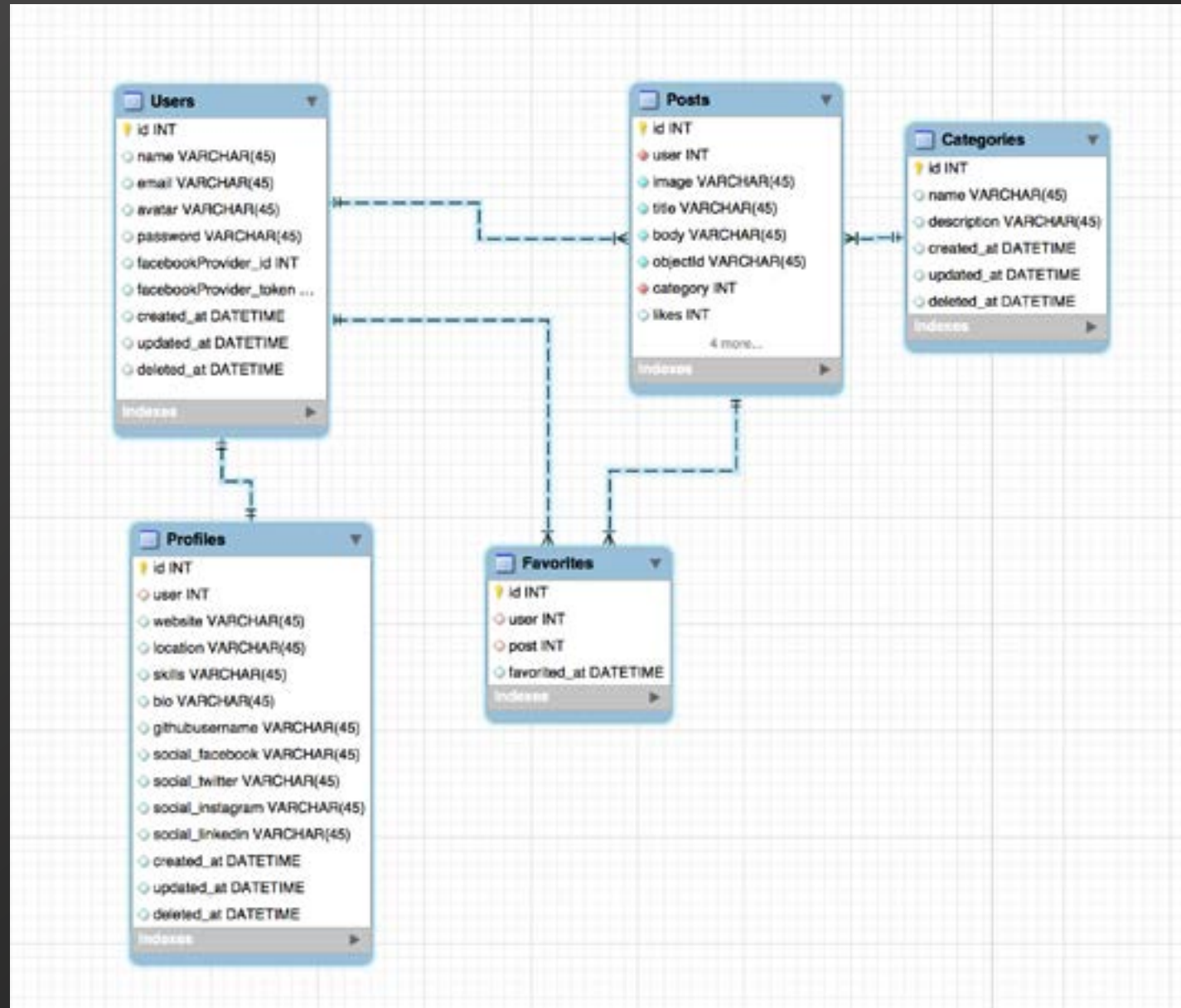


**React**  
Client

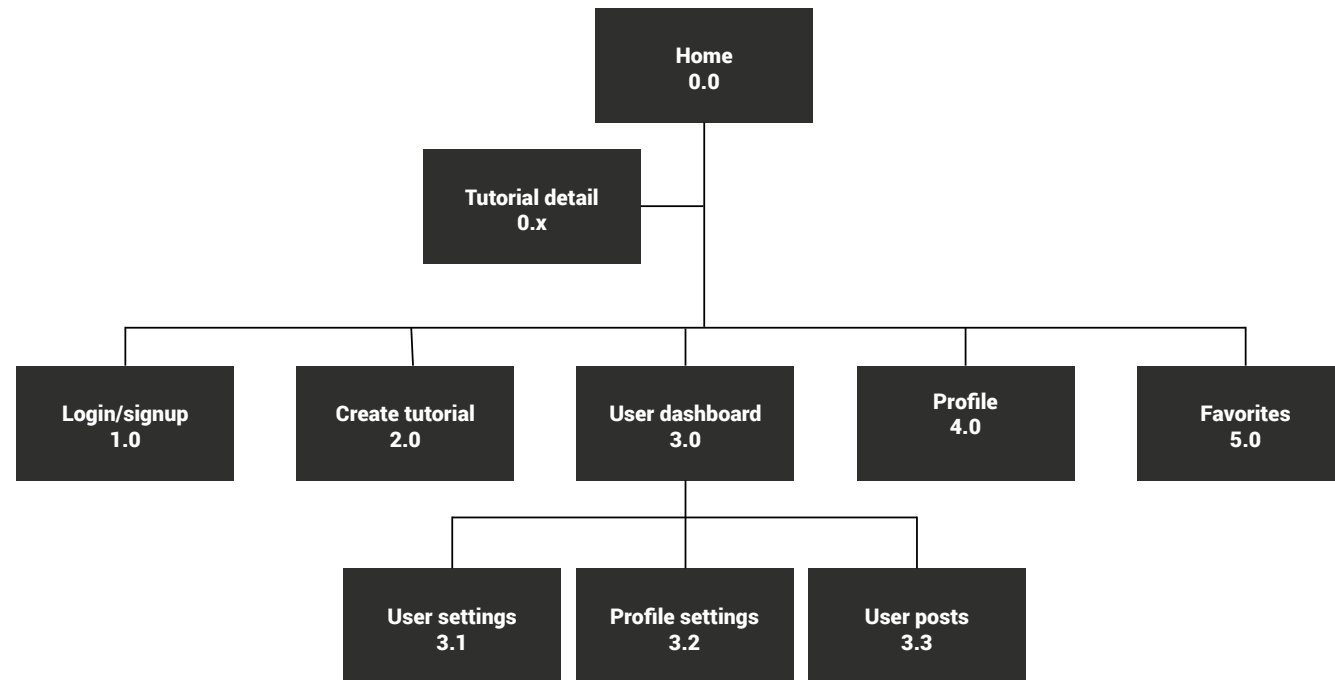
# Features

- **WYSIWYG editor**
- **Algolia search**
- **File upload**

# > Databasemodel



# > Sitemap



# > Wireframes

## CODEKINGS

---

Log in

 Login with facebook

Don't have an account yet? [Sign up](#)

## CODEKINGS

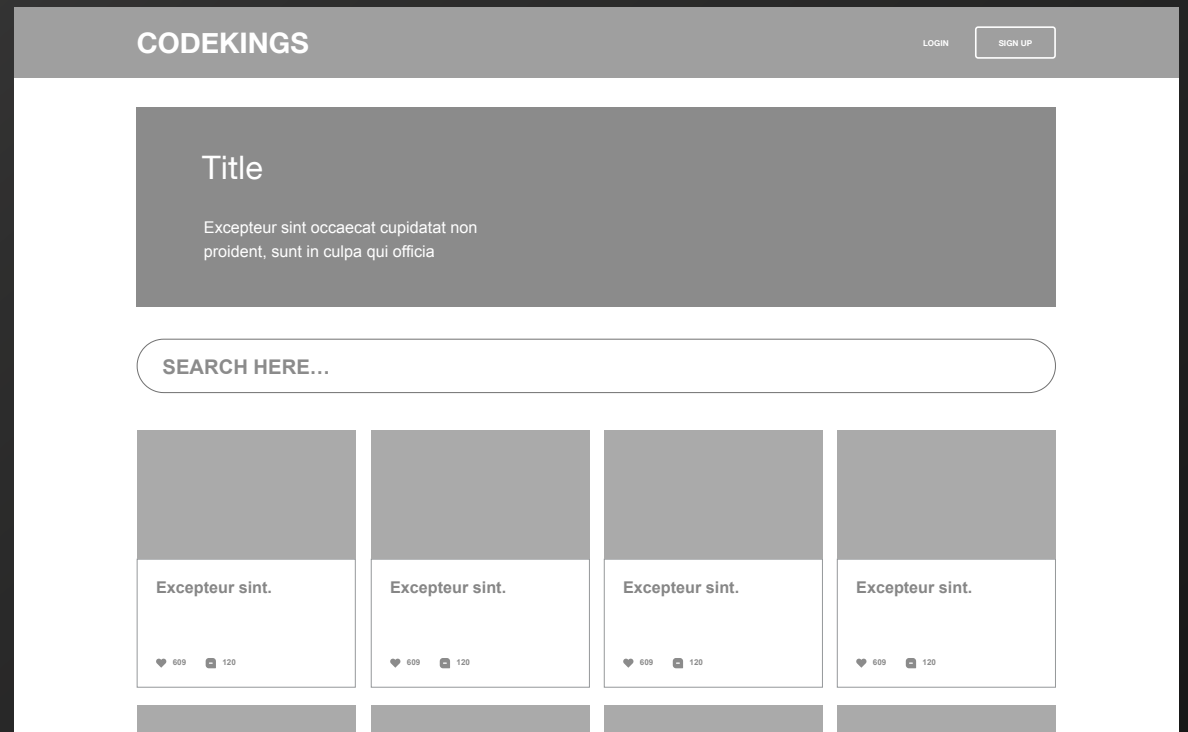
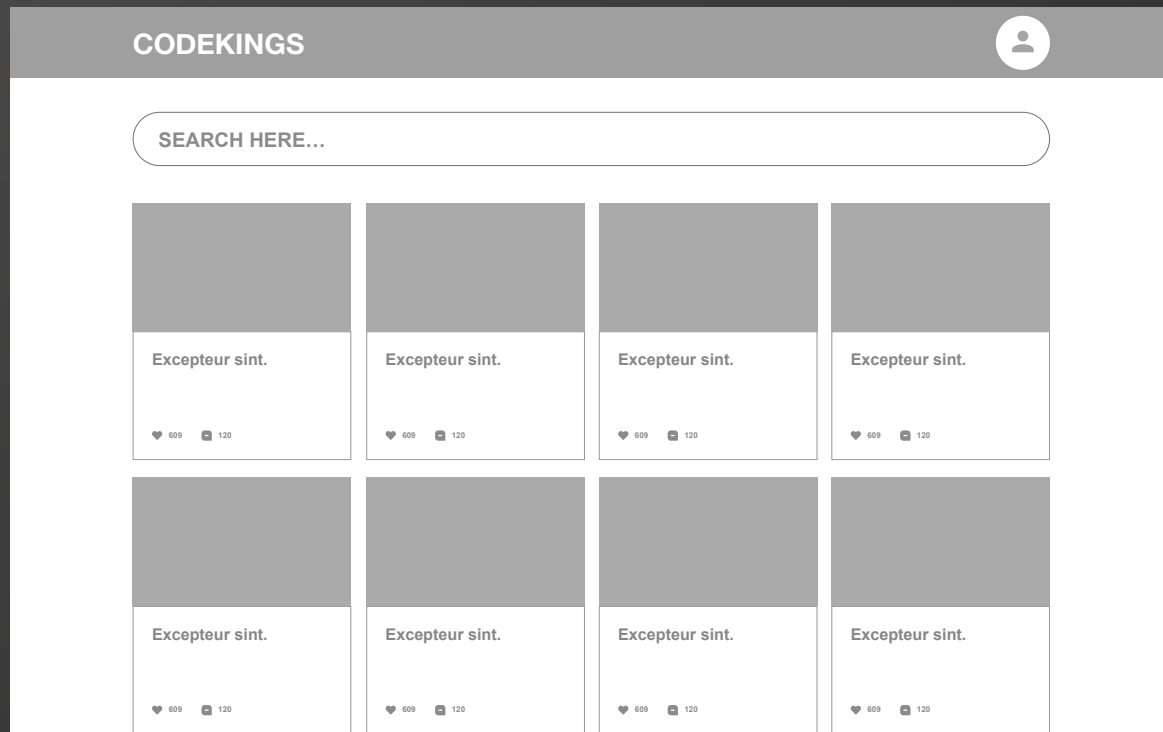
---

Sign up

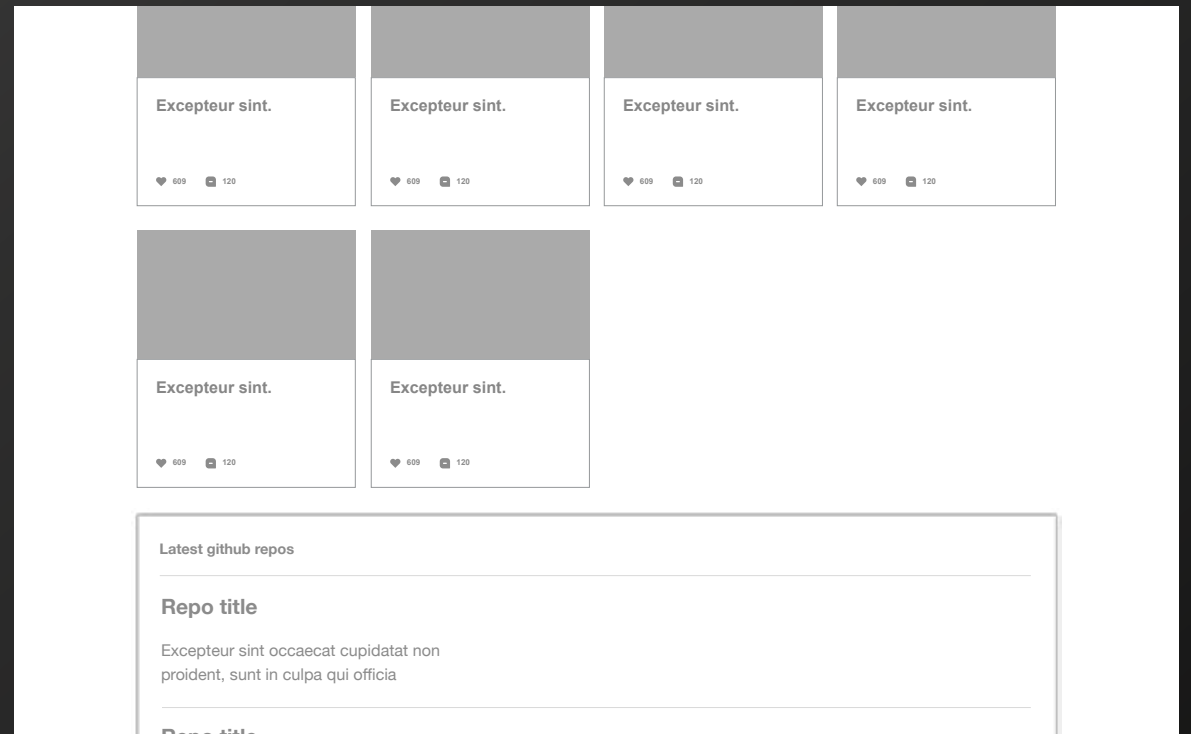
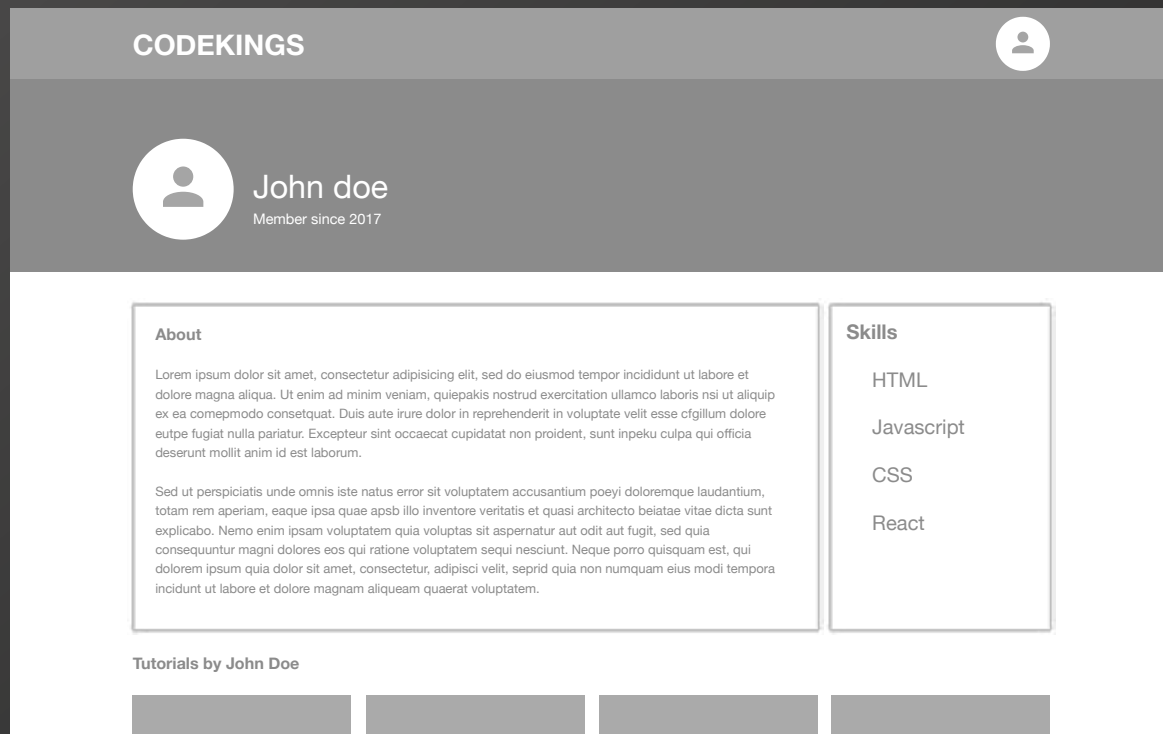
Already have an account? [Log in](#)



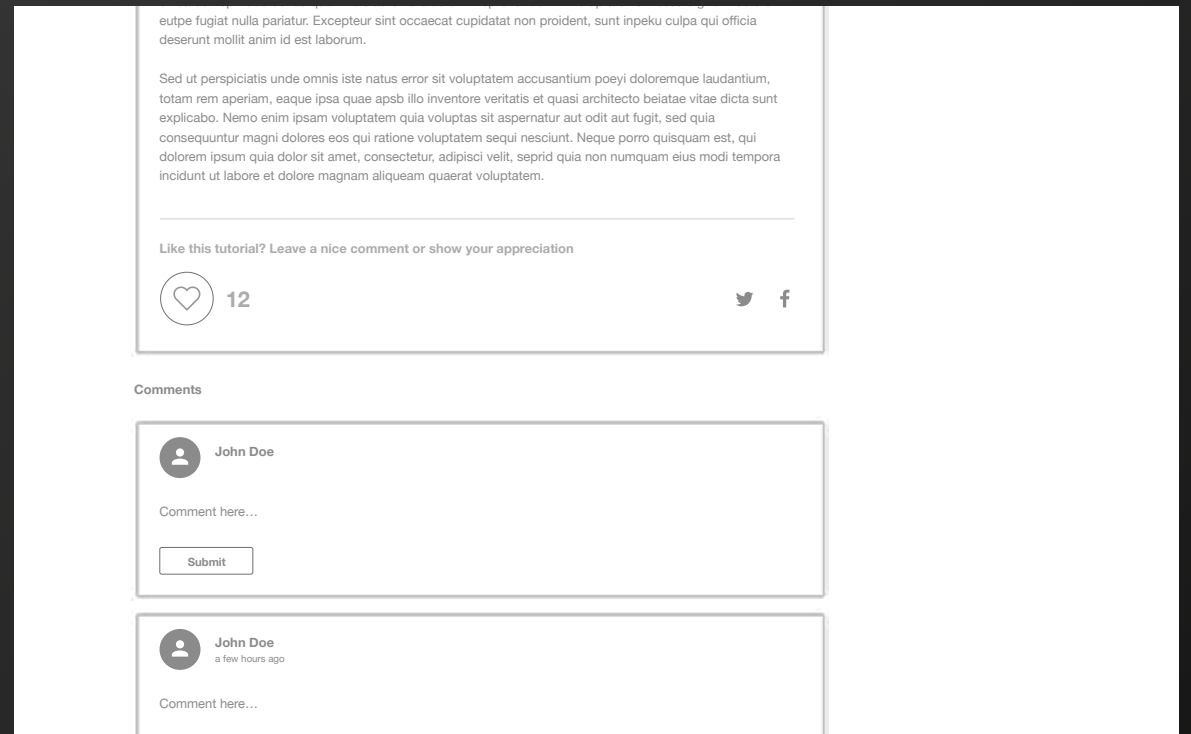
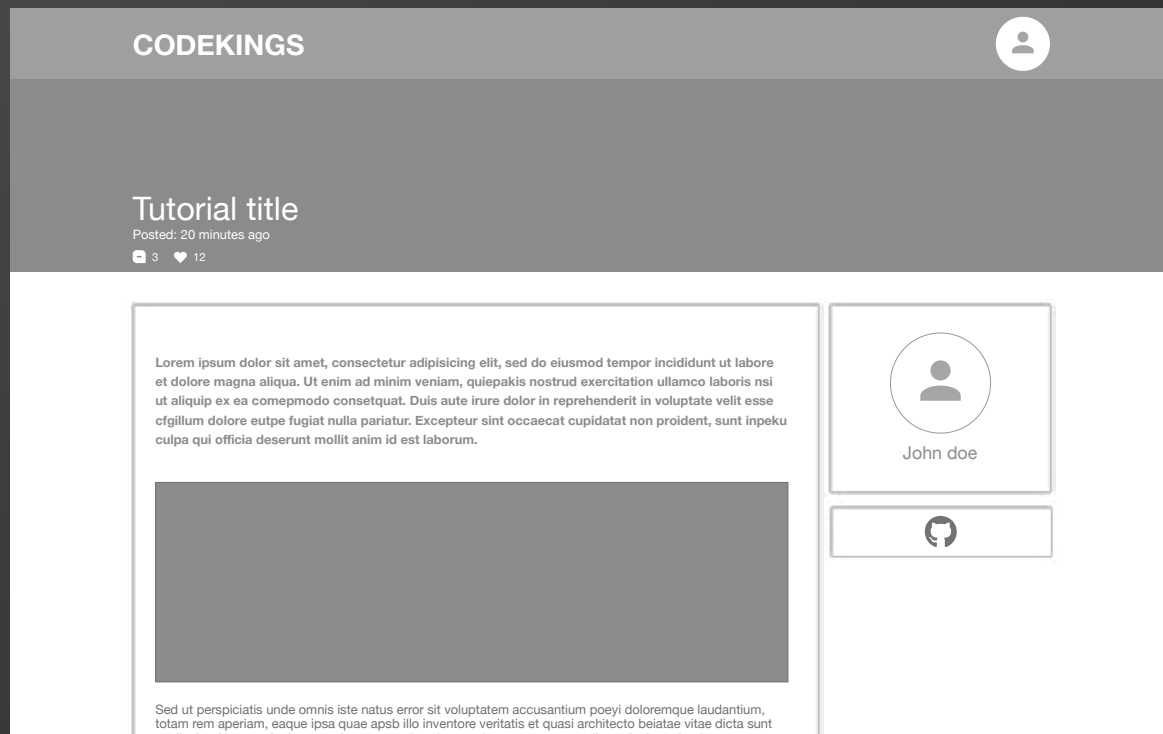
# > Wireframes



# > Wireframes




# > Wireframes



# > Wireframes

CODEKINGS




Create a tutorial

Title

Category

Tutorial content

CODEKINGS



User settings

Profile settings

My posts

User settings

Email

Email

Old password

Old password

New password

new password

Repeat new password

Repeat new password

Save

Delete account

# > Wireframes

CODEKINGS

User settingsProfile settingsMy posts

Profile settings

Website

website

Location

Location

Company

Company

Skills

Skills

Github username

Github username

Bio

Social links

Facebook

facebook url

Twitter

Twitter profile url

Instagram

Instagram profile url

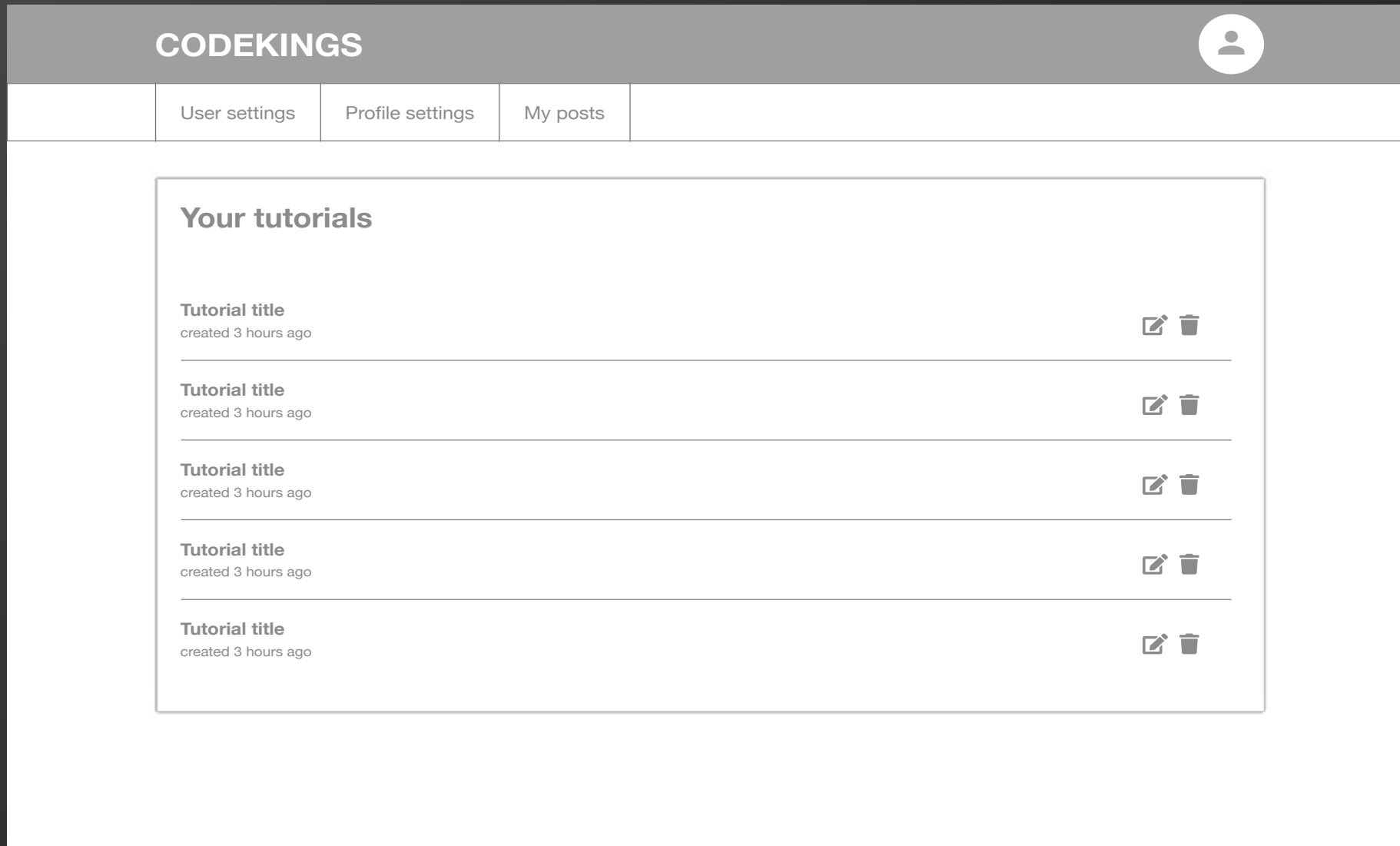
LinkedIn

LinkedIn profile url

Youtube profile url

Youtube profile url

# > Wireframes



# <CODE>KINGS>

## Doel

Het creëren van een maatschappelijk relevante webapplicatie aan de hand van de opgegeven webtechnologieën: React voor de frontend, Node.js voor de backend en mongodb als database.

## Synopsis

**Codekings** is een community gedreven platform met tutorials gemaakt voor liefhebbers van coderen. Van beginner tot code-guru. Iedereen kan vinden wat hij/zij wil leren of ontdekken.

De webapp biedt een uitgebreide keuze aan tutorials die door de gebruikers zelfs gecreëerd kunnen worden.

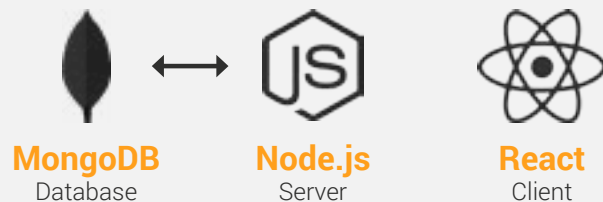
## Resultaat

Het eindresultaat is een webapplicatie waarbij zowel niet geregistreerde bezoekers als leden op zoek kunnen gaan naar de gewenste tutorials.

Op de homepagina is er een overzicht van alle tutorials. Aan de hand van de zoekfunctie kan men snel en efficiënt op zoek gaan naar tutorials gefilterd op datum, titel, auteur of categorie.

Geregistreerde gebruikers hebben de mogelijkheid om zelf tutorials te creëren.

Tenslotte kunnen ze na registratie een profiel aanmaken, waarop getoond wordt wat ze kennen en kunnen.



COLORS

#2F2F2F

#FF9F1C

#E71D36

#E0E0E0

#F0F2EF

#F5F5F5

TYPOGRAPHY

Open Sans

Links	Bold 16px
Body	Regular 16 px
Subheader	Bold 16px
info	Light 12px
Button	Bold 16px

Line height

Raleway

Raleway

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quiepakis nostrud exercitation ullamco laboris nsi ut aliquip ex ea comeprmodo consetquat. Duis aute irure dolor in reprehenderit in voluptate velit esse cggillum dolore eutpe fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt inpeku culpa qui officia deserunt mollit anim id est laborum.

BUTTONS

Filled

Normal

Button

Button

Hover

Button

Button

Normal

Button

Button

Hover

Button

Button

FORMS

Button

Button

Button

Button

FORMS

Label

Placeholder

warning

Label

Placeholder

Label

Placeholder

Save

ICONS



HEADERS

Main

<CODE>KINGS>

Login

Sign up

Sub

Menu item

Menu item

Menu item



# > Code snippets

```
1  const mongoose = require("mongoose");
2  const Schema = mongoose.Schema;
3  const mongoose = require('mongoose').default;
4
5  const PostSchema = new Schema({
6    user: {
7      type: Schema.Types.ObjectId,
8      ref: "users",
9    },
10    image: {
11      type: String,
12    },
13    title: {
14      type: String,
15      required: true,
16    },
17    body: {
18      type: {},
19      required: true,
20    },
21    name: {
22      type: String,
23    },
24    objectId: {
25      type: String,
26    },
27    category: {
28      type: Schema.Types.ObjectId,
29      ref: 'categories'
30    },
31    avatar: {
32      type: String,
33    },
34  });
```

# > Code snippets

```
if (!isValid) {
  return res.status(400).json(errors);
}

User.findOne({ email: req.body.email }).then(user => {
  if (user) {
    return res.status(400).json({ email: "Email already exists" });
  } else {
    const newUser = new User({
      name: req.body.name,
      email: req.body.email.toLowerCase(),
      password: req.body.password,
      password2: req.body.password2
    });

    bcrypt.genSalt(10, (err, salt) => {
      bcrypt.hash(newUser.password, salt, (err, hash) => {
        if (err) throw err;
        newUser.password = hash;
        newUser.save()
          .then(user => {
            const newProfile = new Profile({
              user: user.id,
              skills: 'No skills yet'
            });
          })
        });
      });
    });
  }
});
```

# > Code snippets

```
.....).catch(err => {
.....  res.status(404).json({nofavorites: 'No favorites found'})
.....})
}

exports.add_favorite = (req, res, next) => {
  Favorite.findOne({post: req.body.postId, user: req.user.id})
    .then(favorite => {
      if(favorite) {
        return res.status(400).json({alreadyFavorite: 'This post is already favorited'})
      }

      const newFavorite = new Favorite({
        post: req.body.postId,
        user: req.user.id
      })

      newFavorite.save().then(favorite => res.json(favorite))
    })
    .catch(err => res.status(404).json({nofavoritefound: 'No favorite found'}));
}

exports.remove_favorite = (req, res) => {
  Favorite.findOne({post: req.params.id, user: req.user.id})
    .then(favorite => {
      if(!favorite) {
        return res
          .status(404)
          .json({ nofavorite: "No favorite found" });
      }
    })
}
```

# > Code snippets

```
componentDidMount() {  
  this.props.getCategories();  
}  
  
onChange(e) {  
  const state = this.state;  
  switch (e.target.name) {  
    case "postHeader":  
      state.postHeader = e.target.files[0];  
      break;  
    case "categories":  
      state.category = e.target.value;  
      break;  
    default:  
      state[e.target.name] = e.target.value;  
  }  
  this.setState(state);  
}  
  
componentWillReceiveProps(nextProps) {  
  if (nextProps.errors) {  
    this.setState({ errors: nextProps.errors });  
  }  
}  
  
onSubmit(e) {  
  e.preventDefault();  
  
  const { title, body, postHeader, category } = this.state;  
  let formData = new FormData();
```

# Live preview

<https://codekings.herokuapp.com>

**Bedankt!**