29-short-lnk-app video

Setting up Link Analytics

- first we are going to be updating our docs by using the updateMany in our terminal
- so lets run mongo first in the terminal

```
meteor:PRIMARY> db.links.updateMany({},{})
```

- **And we** know that it takes two object arguments, we want to query all of our docs, so we are going to be leaving the first object empty, on the second argument is what we want to do, in our case we want to $set {visitedCount : 0, lastVisitedAt: null }

```
meteor:PRIMARY> db.links.updateMany({},{$set:{visitedCount: 0, lastVisitedAt: null}})
{ "acknowledged" : true, "matchedCount" : 5, "modifiedCount" : 5 }
```

- and lets run our find() again to reveal our changes

```
meteor:PRIMARY> db.links.find()
{ "_id" : "xbkPucvStF7So8eDh", "url" : "http://google.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : false, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "Hy5cycia-", "url" : "http://yahoo.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : true, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "rkSTq5iT-", "url" : "http://reddit.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : true, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "Hym96xa6b", "url" : "http://fightnews.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : true, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "HJinhOJRZ", "url" : "http://cnet.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : true, "visitedCount" : 0, "lastVisitedAt" : null }
```

- now all of our docs have those two properties
- now lets go to links.js and we are going to be adding those exact properties for new links that get created in the future, so over in Links.insert

```
        Links.insert({
            _id: shortid.generate(),
            url,
            userId: this.userId,
            visible: true,
            visitedCount: 0,
            lastVisitedAt: null
        });
    },
```

- to test this, lets go to the browser and add a brand new link
- and lets check it in the terminal was once

```
meteor:PRIMARY> db.links.find()
{ "_id" : "xbkPucvStF7So8eDh", "url" : "http://google.com", "userId" : "afiuMxY3bEC8nYWTq", "vis
ible" : false, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "Hy5cycia-", "url" : "http://yahoo.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : t
rue, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "rkSTq5iT-", "url" : "http://reddit.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" :
true, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "Hym96xa6b", "url" : "http://fightnews.com", "userId" : "afiuMxY3bEC8nYWTq", "visible"
 : true, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "HJinhOJRZ", "url" : "http://cnet.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" : tr
ue, "visitedCount" : 0, "lastVisitedAt" : null }
{ "_id" : "S1XefqyA-", "url" : "http://google.com", "userId" : "afiuMxY3bEC8nYWTq", "visible" :
true, "visitedCount" : 0, "lastVisitedAt" : null }
```

- now we are going to figure out how to track a visit
- and we are going to use a method at the exact time some one gets redirected and that happens in our server main.js
- we are going to use Meteor.call
- and we are going to call a method called links.trackVisit that we still havent created and all we need to pass in is our _id that we are trying to track

```
import { Meteor } from 'meteor/meteor';
import { WebApp }from 'meteor/webapp';

import '../imports/api/users';
import { Links } from '../imports/api/links';
import '../imports/startup/simple-schema-configuration.js';

Meteor.startup(() => {
    WebApp.connectHandlers.use((req,res, next)=>{
        const _id = req.url.slice(1);
        const link = Links.findOne({_id});

        if(link){
            res.statusCode = 302;
            res.setHeader('Location', link.url);
            res.end();
            Meteor.call('links.trackVisit', _id);
        }else{
            next();
        }
    });
});
```

- now lets go to links.js to define that method we just created
- we are going to be creating that method down below, and we going to validate the _id by using once again SimpleSchema

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';
import SimpleSchema from 'simpl-schema';
import shortid from 'shortid';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
    'links.insert'(url){
        if(!this.userId){
            throw new Meteor.Error('not-authorized')
        }
        new SimpleSchema({
            url:{
            type: String,
            label:'Your link',
            regEx: SimpleSchema.RegEx.Url
            }
        }).validate({ url });

        Links.insert({
            _id: shortid.generate(),
            url,
            userId: this.userId,
            visible: true,
            visitedCount: 0,
            lastVisitedAt: null
        });
    },
    'links.setVisibility'(_id, visible){
        //check if user is logged in. Throw an error if not.
        if(!this.userId){
            throw new Meteor.Error('not-authorized');
        }
        new SimpleSchema({
            _id: {
                type: String,
                min: 1
            },
            visible:{
                type: Boolean
            }
        }).validate({_id, visible});

        Links.update({
            _id,
            userId: this.userId
        },{
            $set:{ visible }
        })
    },
    'links.trackVisit'(_id){
        new SimpleSchema({
            _id: {
                type: String,
                min: 1
            }
        }).validate({_id});
    }
});
```

- and now are going to update a single document
- and we know it takes two object arguments , the first will be targeting the _id and the second argument will be what we want to do
- we are going to use $set and we want to increment
- for lastVisitedAt we going to give it the value of new Date constructor and are getting the time getTime()

```
    'links.trackVisit'(_id){
        new SimpleSchema({
            _id: {
                type: String,
                min: 1
            }
        }).validate({_id});
        Links.update({_id},{
            $set: {
                lastVisitedAt: new Date().getTime()
            }
        });
    }
});
```

- we next are going to be using another Mongodb operator called $inc which is short for increment, it allows you to increment a numeric property without know what its value was
- and $inc property is also an object and it takes what you what you want to increment and by how much, in our case is by 1, there is no $dec for decrement so if we want to decrement we have to use -1

```
import { Mongo } from 'meteor/mongo';
import { Meteor } from 'meteor/meteor';
import SimpleSchema from 'simpl-schema';
import shortid from 'shortid';

export const Links = new Mongo.Collection('links');

if (Meteor.isServer){
    Meteor.publish('links', function(){
        return Links.find({userId : this.userId});
    })
}
Meteor.methods({
    'links.insert'(url){
        if(!this.userId){
            throw new Meteor.Error('not-authorized')
        }
        new SimpleSchema({
            url:{
            type: String,
            label:'Your link',
            regEx: SimpleSchema.RegEx.Url
            }
        }).validate({ url });

        Links.insert({
            _id: shortid.generate(),
            url,
            userId: this.userId,
            visible: true,
            visitedCount: 0,
            lastVisitedAt: null
        });
    },
    'links.setVisibility'(_id, visible){
        //check if user is logged in. Throw an error if not.
        if(!this.userId){
            throw new Meteor.Error('not-authorized');
        }
        new SimpleSchema({
            _id: {
                type: String,
                min: 1
            },
            visible:{
                type: Boolean
            }
        }).validate({_id, visible});

        Links.update({
            _id,
            userId: this.userId
        },{
            $set:{ visible }
        })
    },
    'links.trackVisit'(_id){
        new SimpleSchema({
            _id: {
                type: String,
                min: 1
            }
        }).validate({_id});
        Links.update({_id},{
            $set: {
                lastVisitedAt: new Date().getTime()
            },
            $inc: {
                visitedCount: 1
            }
        });
    }
});
```

- now lets go to LinksListItem to set the PropType definition
- lastVisitedAt was set to witout the isRequired because it is defined a null, so giving it a isRequired would cause an error

```
LinksListItem.propTypes = {
    _id: React.PropTypes.string.isRequired,
    url: React.PropTypes.string.isRequired,
    userId: React.PropTypes.string.isRequired,
    visible: React.PropTypes.bool.isRequired,
    shortUrl: React.PropTypes.string.isRequired,
    visitedCount: React.PropTypes.number.isRequired,
    lastVisitedAt: React.PropTypes.number
}
```

- and now up above lets add those two properties

```
import React from 'react';
import Clipboard from 'clipboard';
import { Meteor } from 'meteor/meteor';

export default class LinksListItem extends React.Component{

    constructor(props){
        super(props);
        this.state = {
            justCopied : false
        }
    }

    componentDidMount(){
        this.clipboard = new Clipboard(this.refs.copy);
        this.clipboard.on('success', ()=>{
            this.setState({justCopied: true})
            setTimeout(()=>{
                this.setState({justCopied: false});
            },1000);
        }).on('error', ()=>{
            alert('please copy the link manually');
        });
    }
    componentWillUnmount(){
        this.clipboard.destroy();
    }
    render(){
        return(
            <div>
                <p>{this.props.url}</p>
                <p>{this.props.shortUrl}</p>
                <p>{this.props.visible.toString()}</p>
                <p>{this.props.visitedCount} - {this.props.lastVisitedAt}</p>
                <button ref="copy" data-clipboard-text={this.props.shortUrl}>
                {this.state.justCopied ? 'Copied' : 'Copy'}
                </button>
                <button onClick={()=>{
                    Meteor.call('links.setVisibility', this.props._id, !this.props.visible);
                }}>{this.props.visible ? 'Hide' : 'Unhide'}</button>
            </div>
        )
    }
}

LinksListItem.propTypes = {
    _id: React.PropTypes.string.isRequired,
    url: React.PropTypes.string.isRequired,
    userId: React.PropTypes.string.isRequired,
    visible: React.PropTypes.bool.isRequired,
    shortUrl: React.PropTypes.string.isRequired,
    visitedCount: React.PropTypes.number.isRequired,
    lastVisitedAt: React.PropTypes.number
}
```

- we should 0 - in the screen because our default is 0 and null
- now lets test by copying one our url links and visiting it in another tab
- and now it should say *1 - 1509038437337*
- this huge number is the time stamp that we are going to be taking care of on the next lesson
-