# Guided Project | Step 2 - Twitter

## Learning goals

After this lesson, you will be able to:

- Visit user's profile from a general timeline
- Follow new users when you are logged in
- Load your timeline, that will contain your own tweets and also tweets from people you are following.

## Introduction

In the previous learning unit we implemented the basic authorization and authentication of our twitter app. We also created all the functionality to be able to publish a tweet and see all our tweets.

In this learning unit we will complete the exercise by adding the features that allow us follow other users and see their timeline.

Let's start coding!

## Public timeline

First of all, we will create a public timeline, to allow **any visitor** to see all the published tweets. Let’s create the controller that will manage this timeline.

We will create a new `timelineController` in the `routes` folder, that will manage the route to show the timeline.

```
 1  // routes/timelineController.js
 2  const express            = require("express");
 3  const timelineController = express.Router();
 4
 5  // Models
 6  const Tweet = require("../models/tweet");
 7
 8  // Moment to format dates
 9  const moment = require("moment");
10
11  timelineController.get("/", (req, res) => {
12    Tweet
13      .find({}, "user_name tweet created_at")
14      .sort({ created_at: -1 })
15      .exec((err, timeline) => {
16        res.render("timeline/index", { timeline, moment });
17    });
18  });
19
20  module.exports = timelineController;
```

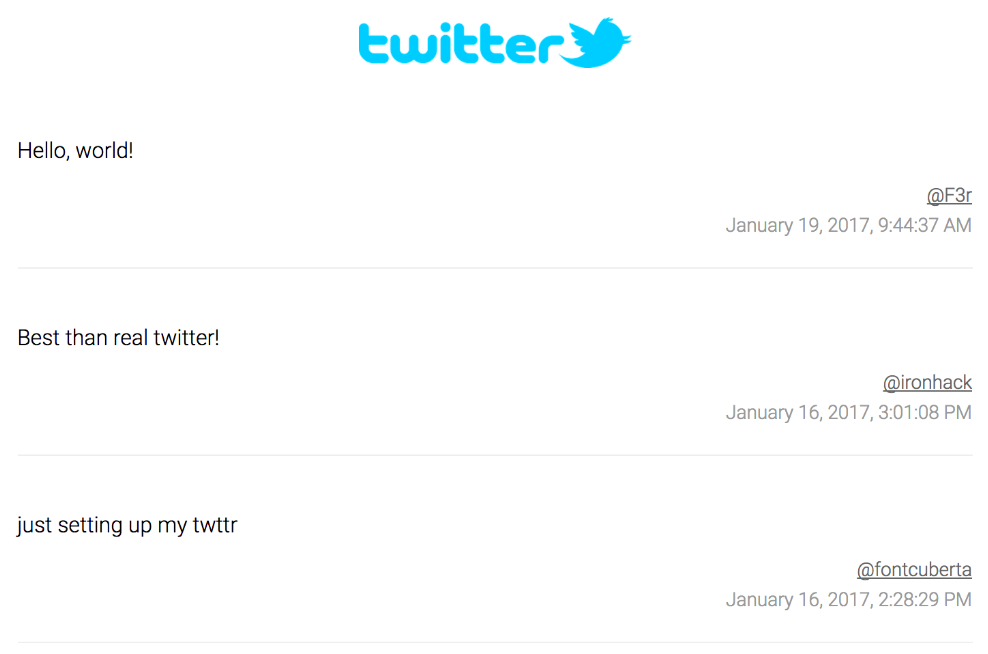
```
// app.js
const timelineController = require("./routes/timelineController");
app.use('/timeline', timelineController);
```

Note how we are including the `Tweet` model and the `Moment` (https://www.npmjs.com/package/moment) package. We use the moment library to format the date in our views. Let’s create the corresponding view.

As you can see in line 16, the view is called `index.ejs` , and its located in the `views/timeline/` path:

```
<!-- views/timeline/index.ejs -->

<div id="timeline">
  <% if (timeline.length > 0) { %>
    <% timeline.forEach(function(tweet) { %>
      <div class="tweet-container">
        <p><%= tweet.tweet %></p>
        <p class="date">
          <span>
            <a href="/profile/<%= tweet.user_name %>">
              @<%= tweet.user_name %>
            </a>
          </span><br>
          <%= moment(tweet.created_at).format("LL, LTS") %>
        </p>
      </div>
    <% }) %>
  <% } else { %>
    <div class="tweet-container no-tweets">
      <p>The timeline is empty</p>
    </div>
  <% } %>
</div>
```

Now we can open the browser and go to `http://localhost:3000/timeline` and we will be able to see the public timeline:


Hello, world!

@F3r
January 19, 2017, 9:44:37 AM

Best than real twitter!

@ironhack
January 16, 2017, 3:01:08 PM

just setting up my twttr

@fontcuberta
January 16, 2017, 2:28:29 PM

> Remember your server should be running to be able to open the page.

Note how the username in the global timeline is a link to a profile page. This is the following step we are going to cover, we will create the profile page.

## Profile page

We are going to create a profile page that is accessible for everyone. That means you will not have to be logged in to see it. We will create `/profile/{username}` as the route for a profile.

Let’s create the `profileController.js` in the `routes` folder:

```
const express           = require("express");
const profileController = express.Router();

// User model
const User  = require("../models/user");
const Tweet = require("../models/tweet");

// Moment to format dates
const moment = require("moment");

profileController.get("/:username", (req, res, next) => {
  User
    .findOne({ username: req.params.username }, "_id username")
    .exec((err, user) => {
      if (!user) { return next(err); }

      Tweet.find({ "user_name": user.username }, "tweet created_at")
        .sort({ created_at: -1 })
        .exec((err, tweets) => {
          res.render("profile/show", {
            username: user.username,
            tweets,
            moment
          });
      });
  });
});
```

Now we have to add the controller in the `app.js` file, and generate the route to access the profiles:

```
// Controllers
const authController     = require("./routes/authController");
const profileController  = require("./routes/profileController");
const tweetsController   = require("./routes/tweetsController");
const timelineController = require("./routes/timelineController");
```

```
// Routes
app.use("/", authController);
app.use("/profile", profileController);
app.use("/timeline", timelineController);
```

Note how we add the `profileController` with `/profile` in the middleware. That allow us to use `get("/:username")` inside the controller to represent the route `/profile/:username`.

To finish the profile page, we have to add the view. We will create it in `/views/profile/show.ejs` file:

```
<div id="profile">
  <h2>@<%= username %></h2>
</div>

<div id="container">
  <% if (tweets.length > 0) { %>
    <% tweets.forEach(function(tweet) { %>
      <div class="tweet-container">
        <p><%= tweet.tweet %></p>
        <p class="date"><%= moment(tweet.created_at).format("LL, LTS") %>
      </div>
    <% }) %>
  <% } else { %>
    <div class="tweet-container no-tweets">
      <p>@<%= username %> hasn't tweeted yet.</p>
    </div>
  <% } %>
</div>
```

We can see the profile page by going to `http://localhost:3000/profile/ironhack`. In this case, we have Ironhack profile, where we can find all the account tweets. We need just one more thing to be able to load our own timeline: following accounts.

## Follow users

Before start coding, let's think a little bit which is the best way to store the following list. One user will be able to follow several users. In NoSQL, we represent this by storing an array in the user model.

So, the first thing we have to do is update the user model to be able to save that array in the database. We have to update it by adding the `following` field to the mongoose Schema:

```
// models/user.js

const userSchema = new Schema({
  username: String,
  password: String,
  following: [{ type: Schema.Types.ObjectId, ref: "User" }]
}, {
  timestamps: {
    createdAt: "created_at",
    updatedAt: "updated_at"
  }
});
```

Next step we have to think about: who are we able to follow? We can’t follow ourselves, of course. We have to show the follow button in other accounts profile, and we have to do that when we are logged in.

First of all let’s change the profile show `GET` route, by adding the following:

```
// tweetsController.js

// other code

Tweet.find({ "user_name": user.username }, "tweet created_at")
  .sort({ created_at: -1 })
  .exec((err, tweets) => {
    res.render("profile/show", {
      tweets,
      moment,
      username: user.username,
      session: req.session.currentUser
    });
});
```

And add the following condition in the view:

```
<!--  views/profile/show.ejs -->

<% if (session !== undefined && username !== session.username) { %>
  <form action="/profile/<%= username %>/follow" method="POST">
    <button class="button blue">Follow</button>
  </form>
<% } %>
```

Great! Now we have to create the route to follow accounts. We want to allow users follow other users when they are logged in. The `/profile/:username/follow` route has to be protected. We will add the following code at the end of the `profileController.js` file:

```
// routes/profileController.js

profileController.use((req, res, next) => {
  if (req.session.currentUser) { next(); }
  else { res.redirect("/login"); }
});

profileController.post("/:username/follow", (req, res) => {
  User.findOne({ "username": req.params.username }, "_id").exec((err, follow) =
    if (err) {
      res.redirect("/profile/" + req.params.username);
      return;
    }

    User
      .findOne({ "username": req.session.currentUser.username })
      .exec((err, currentUser) => {
        var followingIndex = currentUser.following.indexOf(follow._id);

        if (followingIndex > -1) {
          currentUser.following.splice(followingIndex, 1)
        } else {
          currentUser.following.push(follow._id);
        }

        currentUser.save((err) => {
          req.session.currentUser = currentUser;
          res.redirect("/profile/" + req.params.username);
        });
      });
  });
});
```

We have implemented the follow/unfollow feature here. If we are following an account, we remove it from the `following` field. On the other hand, if we are not following it, we have to add it to the array.

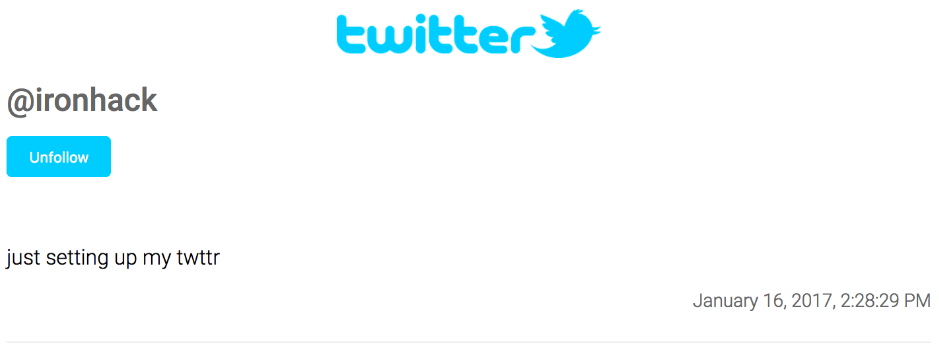
We have to do exactly the same in the view. If we are following the account, we have to show the "Unfollow" button, and viceversa. We update the controller as follows:

```
// profileController.js

// other code


if (req.session.currentUser) {
  isFollowing = req.session.currentUser.following.indexOf(user._id.toStri
}

Tweet.find({ "user_name": user.username }, "tweet created_at")
  .sort({ created_at: -1 })
  .exec((err, tweets) => {
    res.render("profile/show", {
      username: user.username,
      tweets,
      moment,
      session: req.session.currentUser,
      button_text: isFollowing ? "Unfollow" : "Follow"
  });
});
```

In the view, we just have to change the follow button and use the new `button_text` property we just added:

```
<!-- views/profile/show.ejs -->

<% if (session !== undefined && username !== session.username) { %>
  <form action="/profile/<%= username %>/follow" method="POST">
    <button class="button blue"><%= button_text %></%%></button>
  </form>
<% } %>
```

We have implemented all what we need to follow an account. Now we can follow the accounts we want:



@ironhack

just setting up my twttr

January 16, 2017, 2:28:29 PM

Let’s do the last part of the exercise: our own timeline.

## My timeline

At the beginning of this learning unit, we created a public timeline, where we can see all the tweets that have been published in our app. Our users want to see tweets from their interest.

Let's create the profile timeline. In this timeline, we will show the tweets from the accounts that a user is following. We will also add the user's tweets.

Let's suppose that the account @ironhack is following @product_hunt and @elon_musk. Ironhack's timeline will contain the tweets from the three accounts: @product_hunt, @elon_musk, and @ironhack.

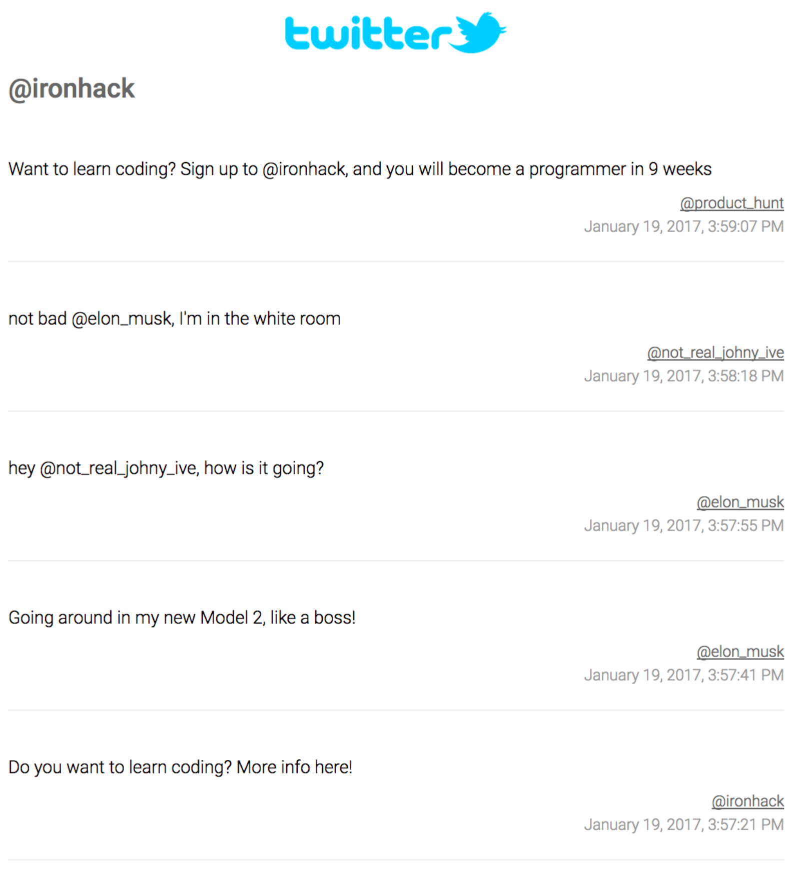
Again, we have to be logged in to access to the private timeline. So we will add the following code in the `profileController.js`:

```
// routes/profileController.js

profileController.get("/:username/timeline", (req, res) => {
  const currentUser = req.session.currentUser;
  currentUser.following.push(currentUser._id);

  Tweet.find({ user_id: { $in: currentUser.following } })
    .sort({ created_at: -1 })
    .exec((err, timeline) => {
      res.render("profile/timeline", {
        username: currentUser.username,
        timeline,
        moment
      });
  });
});
```

This will create our own private timeline. The last step of the exercise is to create the view and load the timeline on it. We will do that in the `views/profile/timeline.ejs` file:

```
<!-- views/profile/timeline.ejs -->

<div id="profile">
  <h2>@<%= username %></h2>
</div>

<div id="timeline">
  <% if (timeline.length > 0) { %>
    <% timeline.forEach(function(tweet) { %>
      <div class="tweet-container">
        <p><%= tweet.tweet %></p>
        <p class="date">
          <span><a href="/profile/<%= tweet.user_name %>">@<%= tweet.user_name
          <%= moment(tweet.created_at).format("LL, LTS") %>
        </p>
      </div>
    <% }) %>
  <% } else { %>
    <div class="tweet-container no-tweets">
      <p>The timeline is empty</p>
    </div>
  <% } %>
</div>
```

The current user will be able to see his timeline by accessing to `http://localhost:3000/profile/:username/timeline`. Here you can find Ironhack's timeline:



@ironhack

Want to learn coding? Sign up to @ironhack, and you will become a programmer in 9 weeks

@product_hunt
January 19, 2017, 3:59:07 PM

not bad @elon_musk, I'm in the white room

@not_real_johny_ive
January 19, 2017, 3:58:18 PM

hey @not_real_johny_ive, how is it going?

@elon_musk
January 19, 2017, 3:57:55 PM

Going around in my new Model 2, like a boss!

@elon_musk
January 19, 2017, 3:57:41 PM

Do you want to learn coding? More info here!

@ironhack
January 19, 2017, 3:57:21 PM

Now, you have your own simple version of twitter. We have implemented some features of the platform, now it's your turn if you want to keep adding more features to it :)

## Summary

In this learning unit we have finished our twitter exercise by adding a few features. We have created a public timeline, where people can see all the published tweets in the platform.

We have also added a new field to the `user` Schema to be able to add accounts you are following. We created an array of `ObjectID` that allow us to add other account's id.

Finally, we created our private timeline to be able to see all the tweets from the accounts we are following as a user.