# Project Anonymous – Connecting art and web development using the JavaScript framework Meteor

Kira Wencek, Computer Science, Art

## Background

The App Idea

- Rob and I have a shared passion for art
- Original plan: collaborative social media site for artists
    - Obviously our idea evolved a lot!
- Modern artists share their works online
- Rob and I thought that anonymity could allow users to judge art without preconceptions
- I wanted to increase my web development knowledge

Why Meteor?

- Full-stack, meaning the front end and back end are coded together in one unified program
- Reactive, meaning changes to the app state/data are reflected to all users in real time

## Learning Outcomes

Wed Development Skills

- Programming with JavaScript and CSS languages
- Used the WebStorm IDE
- A better understanding of the framework Meteor
- A better understanding of a MongoDB database
- How to collaborate and develop using the version control system Git
- Used gitlab.com
- How to design a logical application with an intuitive user interface

## Mutual Leaning Outcomes

Collaboration

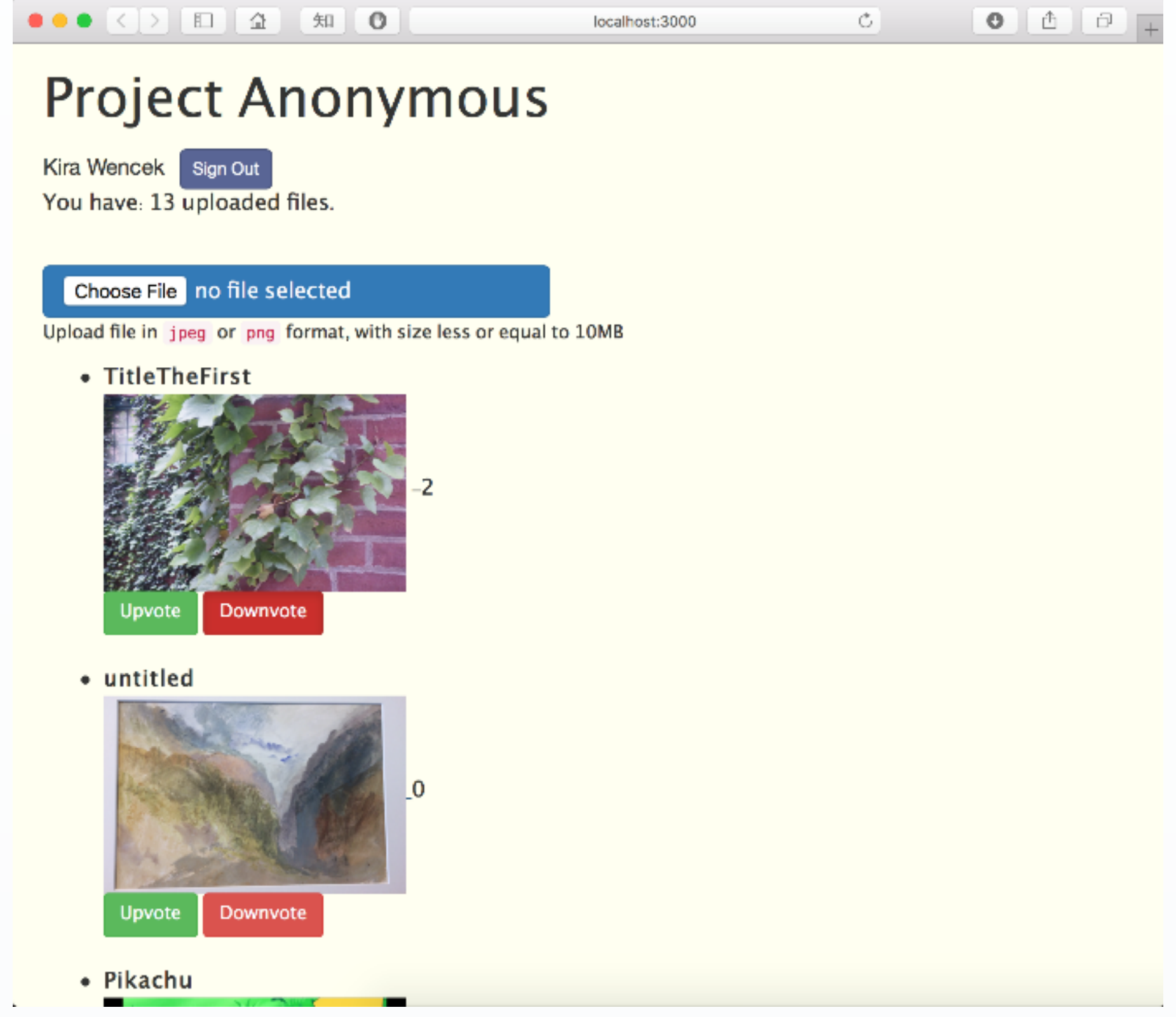
- Rob and I met in person 3 times per week
- Worked together to integrate theory into the app
- Held brainstorm sessions to determine app functionality and features

Communication

- Kept detailed minutes at app functionality meetings
- Explain and convey our respective studies with the others knowledge base in mind
- Regularly updated each other as to current status of project, as well as future plans

## Approach

I used many components to design a web app with Meteor



Project Anonymous

Kira Wencek

You have: 13 uploaded files.

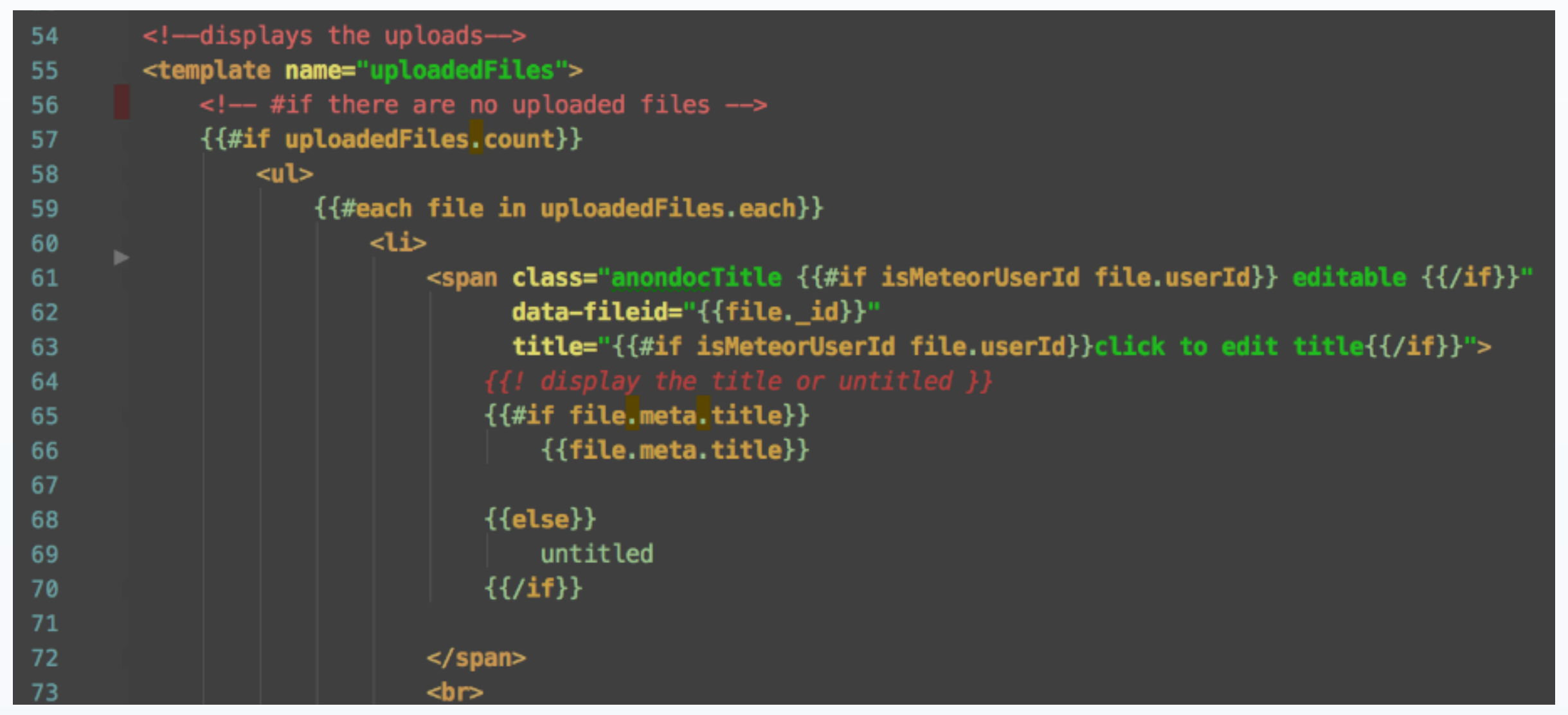
no file selected

Upload file in jpeg or png format, with size less or equal to 10MB

- TitleTheFirst 2
- untitled 0
- Pikachu

A screenshot of the app's current state.

- JavaScript templates provide logic and data to HTML templates with the same name
- HTML templates use spacebars syntax to display that data



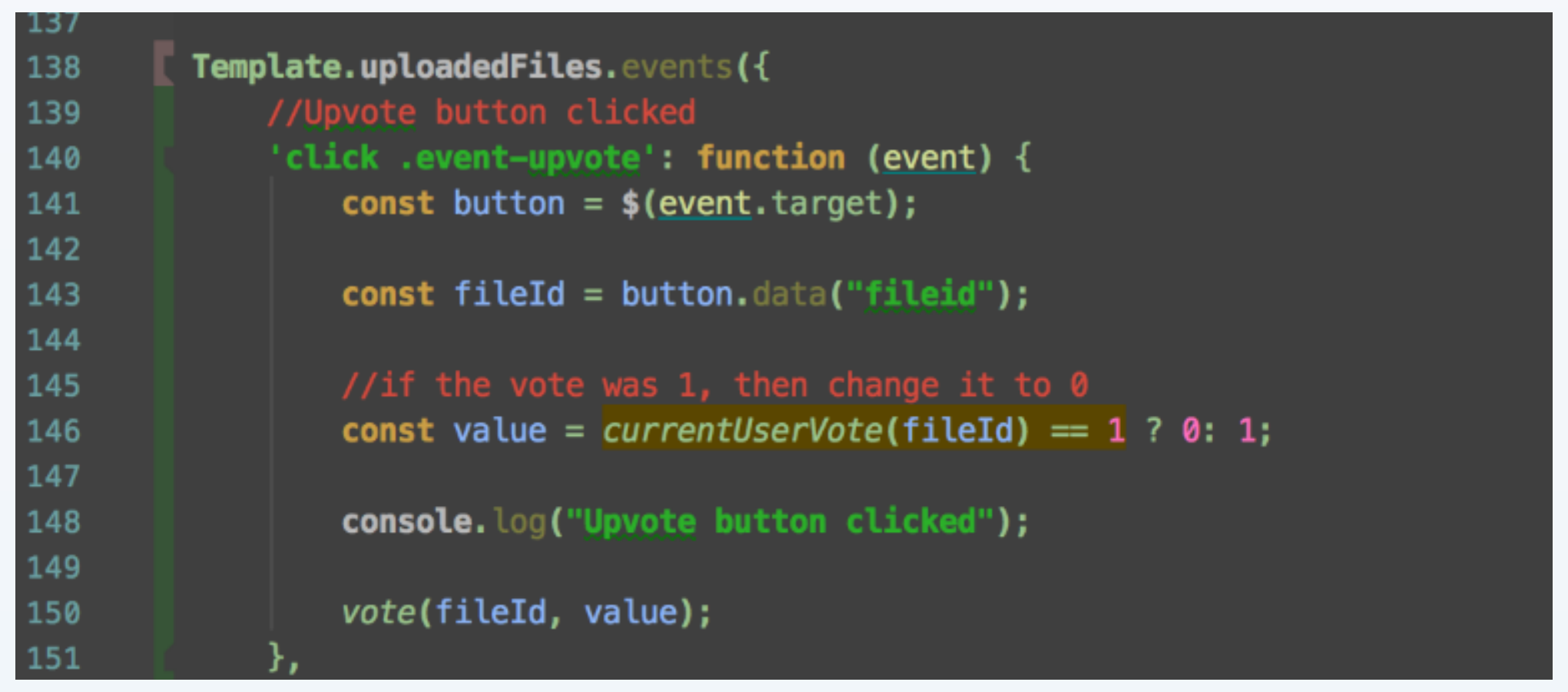
```
<!--displays the uploads-->
<template name="uploadedFiles">
    <!-- #if there are no uploaded files -->
    {{#if uploadedFiles.count}}
        <ul>
            {{#each file in uploadedFiles.each}}
                <li>
                    <span class="anondocTitle {{#if isMeteorUserId file.userId}} editable {{/if}}"
                          data-fileid="{{file._id}}"
                          title="{{#if isMeteorUserId file.userId}}click to edit title{{/if}}">
                        {{! display the title or untitled }}
                        {{#if file.meta.title}}
                            {{file.meta.title}}

                        {{else}}
                            untitled
                        {{/if}}

                    </span>
                    <br>
```

Part of an HTML template for the uploaded files. The {{ }} denote spacebars syntax.

- Event and template helpers



```
Template.uploadedFiles.events({
    //Upvote button clicked
    'click .event-upvote': function (event) {
        const button = $(event.target);

        const fileId = button.data("fileid");

        //if the vote was 1, then change it to 0
        const value = currentUserVote(fileId) == 1 ? 0: 1;

        console.log("Upvote button clicked");

        vote(fileId, value);
    },
```

JavaScript code for the click event helper function for the up-vote button.

- Publish and subscribe
- Developer packages add functionality to Meteor
    - Image upload uses the ostrio:files package
- MongoDB has collections of data. Each piece of data in a collection is called a document
- Setting permissions – for example in this application, only the person who uploaded an image can edit its title

## Discussion

Designing a web application (and programming in general) takes longer than expected!

- This project was originally projected to take one semester
- Much of the first semester consisted of learning how Meteor worked
    - I created two additional test apps in the course of this project
- Even after two semesters of work I have implemented several essential features of the application:
    - Log-in uses Google accounts so no personal information is stored by the application
    - Logged-in users can upload images
    - Logged-in users can change the title of images they uploaded (but not those of other user's images)
    - Any visitor can vote "up" or vote "down" each image
    - Visitors can change their votes -- even users who didn't log in with Google
    - The home page displays a list of thumbnails of uploaded images, including their title and cumulative vote count

Meteor had a big learning curve.

- It was not as intuitive as I had hoped
- My advisor and I were learning together
- I was a little set back right from the start, because I was not very familiar with JavaScript
- Difficult to get a solid understanding of the relationships between imports / JavaScript templates / JavaScript functions / HTML templates / spacebars / CSS
- Understanding the differences between the client and server is an ongoing learning process
- Meteor takes care of a lot behind the scenes
    - E.g. Calls to the server, integration of 3rd-party user accounts
- Adding layers of abstraction can break the app's reactivity
- Git is an extremely helpful development tool, but it also had a leaning curve and numerous kinks to work out throughout the duration of this project

I have significantly furthered my web development skills and knowledge!

- I now feel comfortable saying I understand and can apply the basics of Meteor, MongoDB, and Git

**Acknowledgements**

Thank you to my sponsor David Brown, Department of Computer Science and Statistics