DATA SERVER (MONGODB)

DAY 3

Phonbopit (Chai) Sahakitchatchawan

— Writer @Devahoy

Chai Phonbopit

https://devahoy.com

Phonbopit

http://bit.ly/sut-slide3


mongoDB

Mongoose (Populate)

```
const mongoose = require('mongoose')
const Schema = mongoose.Schema

let Player = new Schema({
  name: String,
  position: String,
  age: Number,
  club: {
    type: Schema.Types.ObjectId,
    ref: 'Club'
  }
})

module.exports = mongoose.model('Player', Player)
```

Mongoose (Populate)

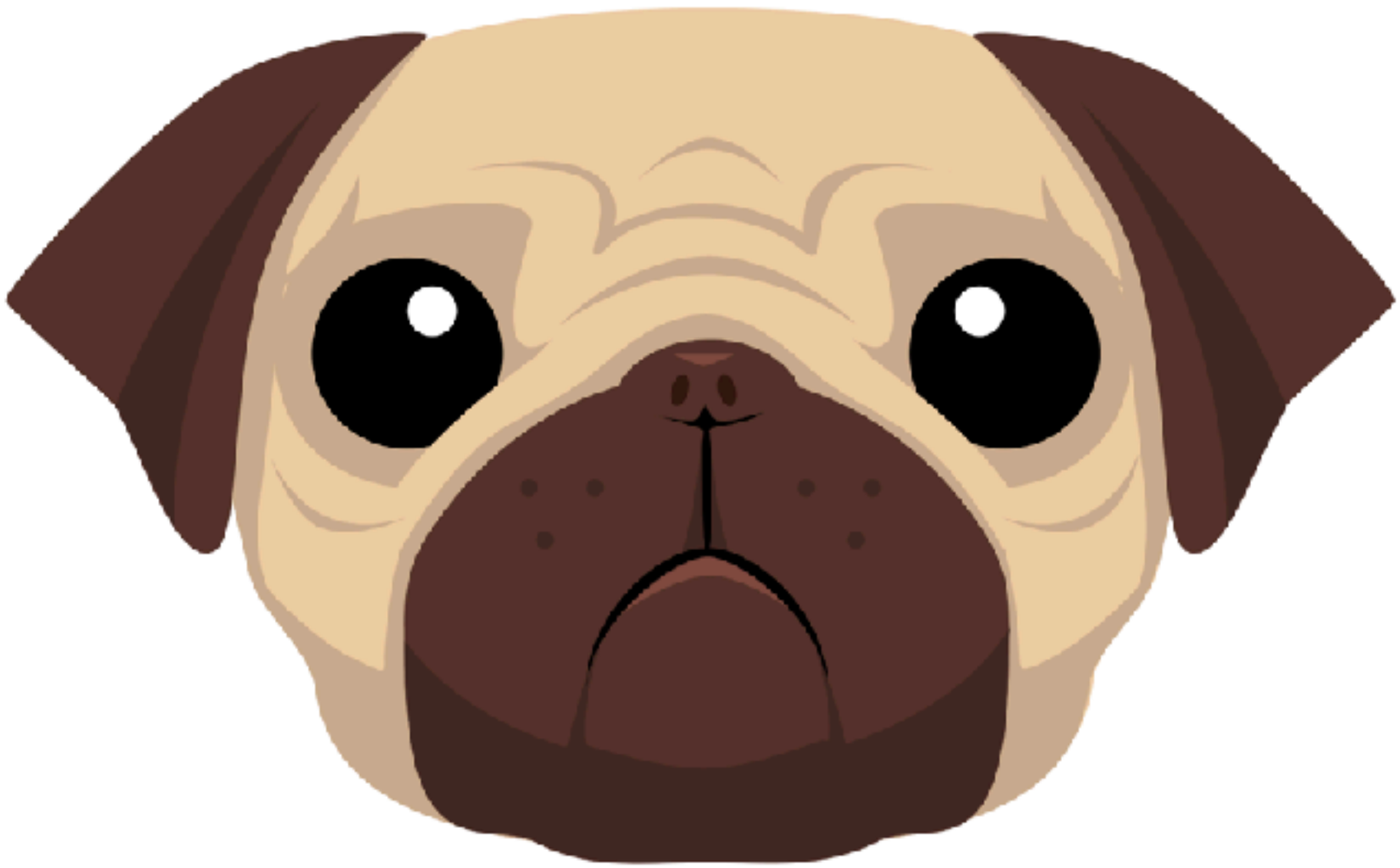
```
Player
   .find()
   .populate('club')
```

```
Player
   .find()
   .populate('club', '_id name')
```

Mongoose (Populate)

```
const Player = require('./player')

router.get('/players', async (req, res) => {
    try {
        const players = await Player.find({}).populate('club')
    } catch (err) {
        res.send('error')
    }
})
```



https://pugjs.org/

http://bit.ly/eventpug

What's Pug?

- Transform to HTML
- Dynamic Code
- DRY (Don't Repeat Yourself

Pug with Express

```
const path = require('path')

app.use(express.static('public'))
app.set('views', path.join(__dirname, 'views'))
app.set('view engine', 'pug')
```

Pug Example

```
// var items = [{ name: 'A }, { name: 'B' }]

// Pug
ul
  each item in items
    li #{item.name}


// HTML
<ul>
  <li>A</li>
  <li>B</li>
</ul>
```

Form (HTML)

```
<form action="/students" method="POST">
  <input type="text" placeholder="Name" name="name">
  <input type="text" placeholder="score" name="score">
  <button type="submit">Create</button>
</form>
```

Form (Pug)

```
form(method='POST')
input(type='text', placeholder='Name', name='name')
input(type='text', placeholder='Score', name='score')
button(type='submit') Create
```

Attributes

```
// Pug
input(
  type='checkbox'
  name='agreement'
  checked
)

// HTML
<input type="checkbox" name="agreement" checked="checked" />
```

Code

```
// item = ['Alice', 'Bob']
// Pug
each item in list
  li
    span= item
    span #{item}
```

```
// HTML
<li>
  <span>Alice</span>
  <span>Bob</span>
</li>
<li>
  <span>Alice</span>
  <span>Bob</span>
</li>
```

Template

```
// layout.pug
doctype html
html
  head
    meta(charset='utf-8')
  body
    block content


// detail.pug
extends layout

block content
  h1 Detail Page
```

Template (event list)

```
extends ../layout

block content
  .event-list
    .columns
  .column.is-12.has-text-centered
    h2 Event List
      a(href='/events/new', class='button is-link button-new') Create new
each event in events
  .columns
    .card.column.is-12
      span #{event.name}
      a(class='button', href=`/events/${event._id}`) Detail
```

Routing (list)

```
router.get('/', async function(req, res) {
  try {
    const events = await Event.find({})
    res.render('events/list', { events })
  } catch (err) {
    res.status(400).send('error')
  }
})
```

Template (event detail)

```
.field.is-grouped.is-grouped-centered
  p.control
    button(class='button is-danger is-delete', data-id=event._id) Delete
  p.control
    a(href=`/events/${event._id}/edit`, class='button is-info') Edit
```

Routing (detail)

```
router.get('/events/:id', async (req, res) => {
  try {
    const { id } = req.params
    const event = await Event.findById(id)
    res.render('events/detail', { event })
  } catch (err) {
    res.status(400).send('error')
  }
})
```

Routing (create)

```
router.post('/events', async (req, res) => {
  try {
    const event = new Event(req.body)
    await event.save()
    res.redirect(`/events/${event._id}`)
  } catch (err) {
    res.status(400).send('error')
  }
})
```

Template (event edit)

```
form(
  method='POST',
  action='/events/#{event._id}',
  class='column is-6 is-offset-3'
)
```

Routing (edit form)

```
router.get('/events/:id/edit', async (req, res) => {
  try {
    const { id } = req.params
    const event = await Event.findById(id)
    res.render('events/edit', { event })
  } catch (err) {
    res.status(400).send('error')
  }
})
```

Routing (edit)

```
// router.put('/events/:id')
router.post('/events/:id', async (req, res) => {
  try {
    const { id } = req.params
    const updated = { $set: req.body }
    const options = { new: true }

   const event = await Event.findByIdAndUpdate(id, updated, options)
   res.render('events/edit', { event, message: 'Updated successfully!' })
  } catch (err) {
    res.status(400).send('error')
  }
})
```

Template (main.js)

```
$(function() {});

$('#selector').on(eventName, callback)
```

```
$.ajax({
  type: 'DELETE',
  url: '/events/' + $(this).attr('data-id'),
}).done(function() {
  window.location.href = '/'
})
```

Routing (Delete)

```
router.delete('/events/:id', async (req, res) => {
  try {
    const { id } = req.params
    await Event.findByIdAndRemove(id)
    res.json({})
  } catch (err) {
    res.status(400).send('error')
  }
})
```

<THANK />