

CookieHour



**Kyra Stephen
Temisan Iwere
Timothy Lui
Simon Aayani
Adam Czyzewski
CSC302**

Learning Goals of Our Project

- Use React in a side project
 - Learn how to use Node
- Learn how to practice TDD with MERN Stack
 - Learn how to use MongoDB as a database

Features We Wanted to get Done

Instructor

- Sync/Export intervals and meetings to personal electronic calendar(s) - P0
- Create office hours interval - P0
- Create new class - P0
- Manage meetings in intervals - P0:
 - Reschedule meetings to another interval
 - Cancel/Delete meetings
 - Move meeting earlier or later within the interval
- Instructor Manage Comments (Comments associated with meeting, shared with student) - P1:
 - Create comment
 - Edit comment
 - Delete comment
 - Find comment
- Delete interval with no meetings - P1
- Instructor Manage notes (Notes private to instructor) - P2:
 - Create note about student meeting
 - Find a note made about a student
 - Find a comment made for a student
 - Delete a note
 - Edit a note
- Generate persistent link to share meetings and/or intervals (eg in a quercus announcement or email) - P2
- Instructor notify system that meetings are running late - P2
- Manage preferences - P3

Student

- Choose meeting slot - P0
- Cancel meeting slot - P0
- Edit meeting slot - P0
- Student Manage Comments P1
 - Create comment
 - Delete, Edit comments
 - Find comments
- Including in advance! (Propose agenda or ask questions) P1
- Efficiently Inform instructor when running late for meeting. System should adapt - P3
- Sync/Export meetings to personal electronic calendar(s) - P3

Features We Actually Completed

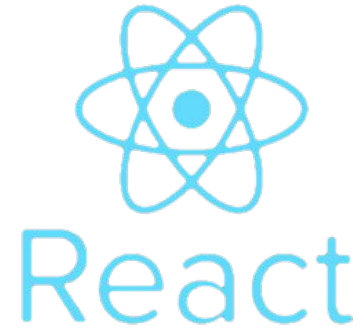
Instructor

- ☒ Sync/Export intervals and meetings to personal electronic calendar(s) - P0
- ☒ Create office hours interval - P0
- ☒ Create new class - P0
- ☒ Manage meetings in intervals - P0:
 - Reschedule meetings to another interval
 - Cancel/Delete meetings
 - Move meeting earlier or later within the interval
- ☒ Instructor Manage Comments (Comments associated with meeting, shared with student) - P1:
 - Create comment
 - Edit comment
 - Delete comment
 - ☒ Find comment
- ☒ Delete interval with no meetings - P1
- ☒ Instructor Manage notes (Notes private to instructor) - P2:
 - Create note about student meeting
 - ☒ Find a note made about a student
 - ☒ Find a comment made for a student
 - Delete a note
 - Edit a note
- ☒ Generate persistent link to share meetings and/or intervals (eg in a quercus announcement or email) - P2
- ☒ Instructor notify system that meetings are running late - P2
- ☒ Manage preferences - P3

Student

- ☒ Choose meeting slot - P0
- ☒ Cancel meeting slot - P0
- ☒ Edit meeting slot - P0
- ☒ Student Manage Comments P1
 - Create comment
 - Delete, Edit comments
 - ☒ Find comments
- ☒ Including in advance! (Propose agenda or ask questions) P1
- ☒ Efficiently Inform instructor when running late for meeting. System should adapt - P3
- ☒ Sync/Export meetings to personal electronic calendar(s) - P3

Technology Stack



Testing Stack



REACT TESTING 
react-testing-library



Testing Example- Backend

```
describe('add meeting with already booked slot', () => {
  it('it should throw "chosen time is already taken" error', (done) => {
    let meeting = {
      "studentEmail": "misan@email.com",
      "teacherEmail": "matt@email.com",
      "date": "19/03/2019",
      "startTime": "13:30",
      "agenda": "confused about A1",
      "courseName": "CSC302"
    };
    let response = {err: "chosen meeting time is already taken"};
    chai.request(server)
      .post('/meeting/create/student')
      .send(meeting)
      .end((err, res) => {
        res.should.have.status(400);
        res.body.should.be.eql(response);
        done();
      });
  });
});
```




REACT TESTING 

react-testing-library

Testing Example - Frontend

```
import { configure, shallow } from 'enzyme';
import Adapter from 'enzyme-adapter-react-16';
import 'jest-enzyme';

configure({ adapter: new Adapter() });

import React from 'react';
import ReactDOM from 'react-dom';
import { render, fireEvent, cleanup } from 'react-testing-library';
import StudentLogin from './StudentLogin';

// automatically unmount and cleanup DOM after the test is finished.
afterEach(cleanup);

describe('Student Login Test', function () {

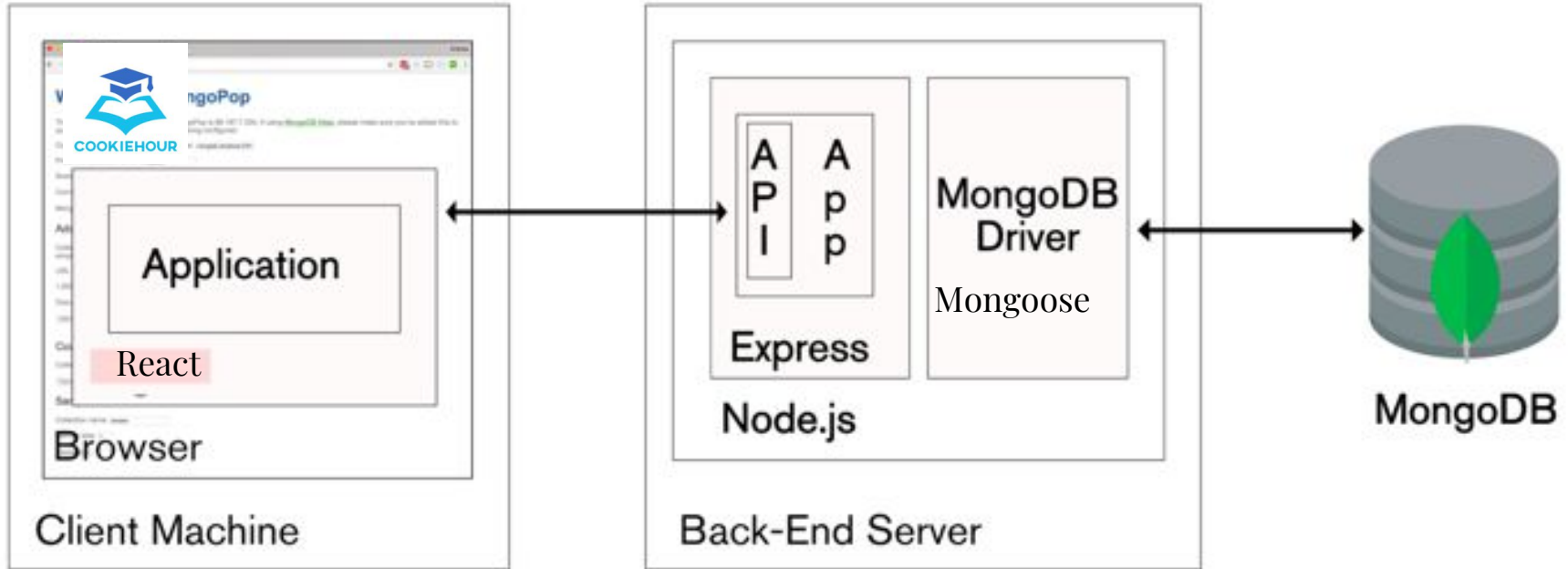
  it('renders without crashing', () => {
    shallow(<StudentLogin />);
  });

  it('username field exists', () => {
    const {getByTestId} = render(
      <StudentLogin />
    );
    expect(getByTestId('email')).not.toBeNull();
  });

  it('password field exists', () => {
    const {getByTestId} = render(
      <StudentLogin />
    );
    expect(getByTestId('password')).not.toBeNull();
  });

});
```


Architecture



Architecture cont - Api.md

user Operations about user

GET /**user** Gets the currently logged in profile of the user

student Operations about student

POST /**student/login** Logs student into the system

GET /**student/profile** Gets the currently logged in student

DELETE /**student/logout/{sessionId}** Logs-out the student out of the system

teacher Operations about teacher

POST /**teacher/login** Logs teacher into the system

GET /**teacher/profile** Gets the current logged in teacher

DELETE /**teacher/logout/{sessionId}** Logs-out the teacher out of the system

meeting Operations about meeting

POST /**meeting/create/student** Creates a meeting for the logged in student

GET /**meeting/get-meeting/{client}** Gets all the meetings for given office hour block

PATCH /**meeting/edit-agenda/student** Edits the agenda of the student for the given meeting

PATCH /**meeting/edit-comment/{client}** Edits comments made in a meeting

PATCH /**meeting/edit-note/teacher** Edits notes made in a meeting

PATCH /**meeting/reschedule-time** Reschedules the time of a meeting

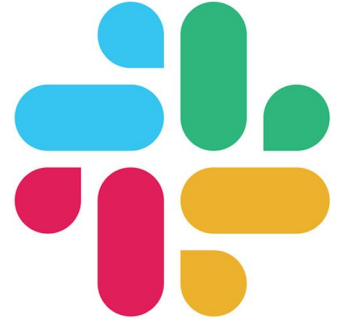
PATCH /**meeting/reschedule-date** Reschedules the date of a meeting

DELETE /**meeting/delete** Unchedules the meeting and frees up the time slot in the office hours

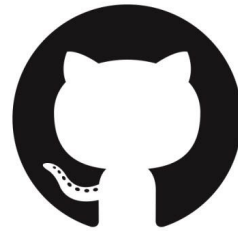
DELETE /**meeting/delete-note/teacher** Deletes a note in a meeting

DELETE /**meeting/delete-comment** Deletes the comment associated with the meeting

How we worked together



Meetings Every
Monday



GitHub

How we worked together

The screenshot displays a Trello board for the 'CookieHour' project. The board is organized into five columns: BackLog, To be taken for Sprint, In Progress, To be reviewed, and Done. Each column contains cards representing tasks, categorized by type (Frontend, Backend, Bug, Testing) and priority (P 1, P 2, P 3). Cards include details like due dates, assignees, and progress indicators. The board also features a top navigation bar with project settings and a right sidebar with additional tools and menu options.

CookieHour ☆ CookieHour Free Public KS AC M SA T 6 Invite

Agile Tools ... Show Menu

BackLog

- Frontend**
Fancier date/time picker
- Frontend**
Instructor can notify to student that meeting is running late
- Frontend**
fix up some UI - css to be more pretty
- Backend**
allow students and instructors to only access app pages when logged in
- Bug**
Verify student is logged in before making edits to meetings
- Frontend**
Student can notify teacher that they

To be taken for Sprint

- Frontend**
add a button to allow teacher/student to export to their calendar
- Testing**
Make existing frontend tests pass
- Backend**
Sync/Export office hours to personal calendar

In Progress

- Backend**
When cookie expires, navigate automatically to sign in page

To be reviewed

- Frontend**
Reschedule/cancel meetings - student
- Backend**
Sync/Export meetings to personal calendar
- Frontend**
Fixed GET and DELETE meeting calls

Done

- Bug**
Delete meeting doesn't work
- Backend**
Get meeting, backend
- Backend** **Bug**
Adding meeting by the student for a 'o'clock' time doesn't work
- Frontend**
student dashboard
- Backend** **Testing**
Allow teacher to delete office hours.
- Frontend**
Allow student to choose meeting slot and write agenda

How we worked together

- Sprint lasted the time to get a phase document done. Average of 2 weeks
 - Story points: 1,2,3,5
- Labels on tasks in trello: Frontend, Backend, Bug, Document
 - Backend – Kyra, Misan , Adam
 - Frontend – Tim, Simon

What went Well

- Finished all tasks with Priority 2 and up
- Learning goal of using MERN stack and practicing TDD achieved
- Frontend and Backend successfully work together

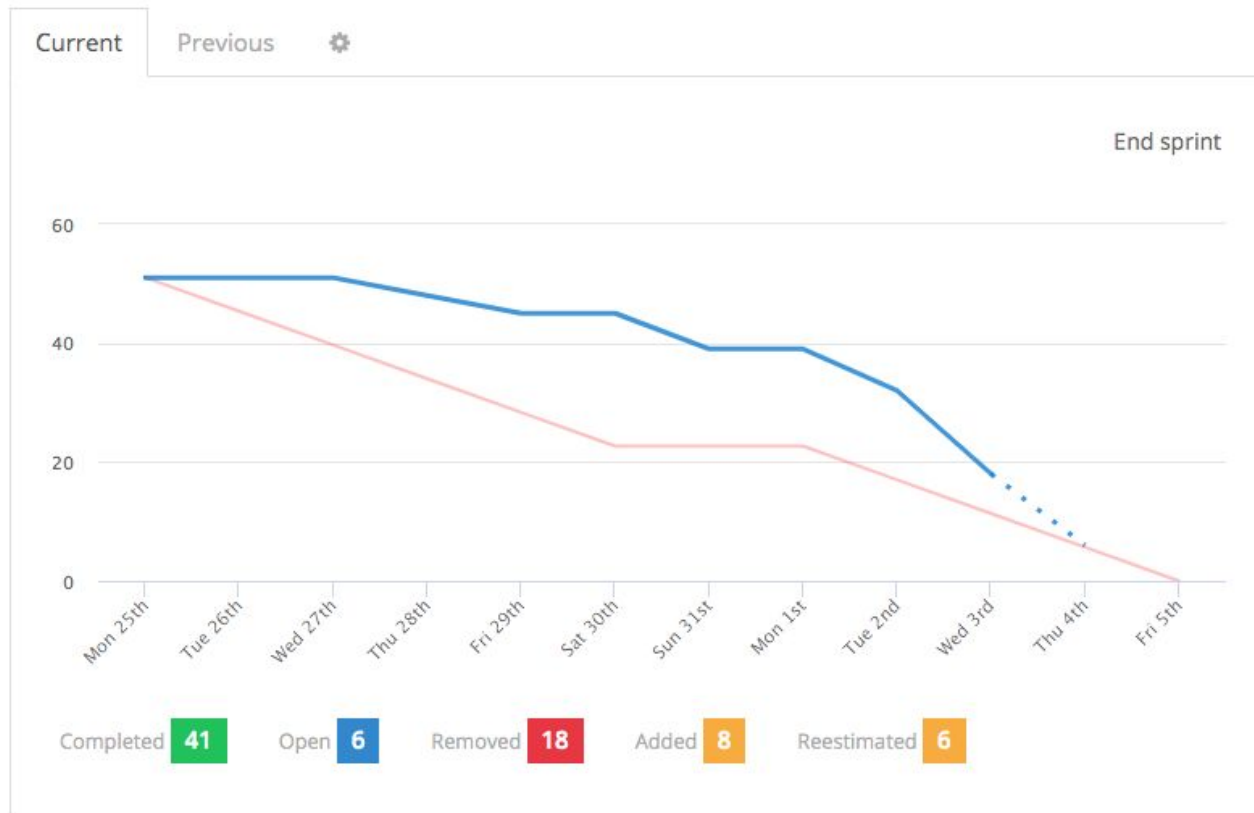
What went Bad

- Frontend testing could be more rigorous
- Didn't get to do notifications feature: "I'm running late" "Meeting is going over the time"
 - Interesting bugs: Some of our functions would falsely throw errors because of mock data in the database not matching up

Phase 2 Burndown



Phase 3 Burndown



Demo



COOKIE HOUR
