Afleveringsopgave

Hacked Hogwarts Student List

This is the final assignment in the Coding Visual Design theme. It is a massive assignment, with a lot of customer-requirements, some even conflicting, so make sure to spend a considerable amount of time planning and structuring your solution.

In class, we will discuss various planning processes, and why some might be preferable to others.

Don't just sit down and solve this assignment in one go - plan it in advance, and take small steps throughout the week!

The backstory (for those who prefer a reason behind the requirements):

You have been hired as the frontender for **Hogwarts School of Witchcraft and Wizardry**, to build a system to help the administrators handle student lists. As test-data you are given the students from the infamous class of 1991.

First, you are just asked to provide an interface to show the list, sort by firstname, lastname, or house, as well as filter by house. Also the interface must provide a "popup" window with detailed information about each student, including photo and house-crest and colors.

As you finish your assignment, you are asked to expand on the solution - the administrator must be able to expel individual students, and see a list of expelled students. On a less dramatic note, two students from each house can be selected as prefects, and this should show in their "popup".

Then the customer experiences a massive shift in political view, and you are bombarded with additional requirements. In addition to prefects, some students can be appointed to join the inquisitorial squad. You are also tasked with implementing racial profiling, by adding "blood-status" to each student - something the original data doesn't have, so you need to devise an algorithm for figuring out a students heritage based on lists of family-names.

You grow a bit tired of these new modifications, so you decide to hack the system. First you want to infiltrate the school, so you "inject" yourself in the list of students (in the

> house of your choice), and make sure that you can't be expelled by the administrator.
>
> Additionally you break the inquisitorial squad by removing students, shortly after the administrator has added them. And finally you mess up your pure-blood algorithm, so it is rather random whether a student is shown as pure-, half-, or muggle-blood.

User requirements

These are the collected specific requirements from the user, note that additional requirements might influence how you should design the solution, so read all of them, before beginning work.

The data

The list of students is to be read from: https://petlatkea.dk/2021/hogwarts/students.json (you can use http or https). The data only contains full names of students, but your solution is expected to split into first, middle, last, and optional nick names. The data is in quite a state, but cannot be expected to be fixed, so your code must clean it properly.

You can download images of the students
from: https://petlatkea.dk/2021/hogwarts/images.zip or use your own, but keep the filenames!

The list

The solution must display a list of students. The list is intended for administrators to get a quick overview of the students in the current year, and sort, filter, and search for certain properties.

Sorting

The administrator must be able to sort the list on at least first, and last names. Also a kind of sort, that would group e.g. students in the same house, or those with certain responsibilities (could be quidditch-players, -captains, prefects, or club memberships).

Filtering

The administrator must be able to filter the view, so only students from one house is shown. Also, filters that only showed those with certain responsibilities, or other properties would be

nice.

Uh, and filtering expelled vs non-expelled students is certainly a requirement!

Searching

The administrator must be able to quickly search for a student, e.g. by first name or last name. A search field, that immediately limits the listed students to those matching the search criteria, would be nice. E.g. search for "ha" would show Harry Potter, Hannah Abbott, Zacharias Smith, and Michael Corner.

About

The interface must show some details about the lists:

- Number of students in each house
- Total number of students (not expelled)
- Number of students expelled
- Number of students currently displayed

Details-popup

The administrator must be able to select a student, and get a "popup" with details.

This popup must be decorated (or "themed" as it is called) with the house crest and colors of the selected student. And must show:

- First name
- Middle name (if any)
- Nick name (if any)
- Last name
- Photo of student (if exists)
- House Crest and colors
- Blood-status
- If the student is:
    - prefect or not
    - expelled or not
    - member of inquisitorial squad or not

Expelling students

The user must be able to expel a student. You decide if the expelling should be done from the list, or from the "popup". Expelling removes a student from the list of students, and adds it to another list of expelled students. Once expelled, a student cannot return to the original list.

Note: Since the JSON-file cannot be modified, no changes will last through a reload. That is okay.

Prefects

Only two students from each house can be selected prefects. Usually a boy and a girl.

The user must be able to make any student a prefect, and also revoke the prefect-status at any time. You must include some sort of system to prevent more than two prefects from each house. It is up to you how to design this. If the user must manually revoke one prefect before creating a new, or if there is simply a confirmation box for this. You can also decide if you want to continue the gender-specific prefects, or allow two boys or two girls.

Blood-status

The system must calculate blood-status for each student. This is an indication of whether the student is from a pure wizarding family, or from a half-wizard, half-muggle family, or just plain muggle.

Use this list: https://petlatkea.dk/2021/hogwarts/families.json that is a list of all known pure-blooded wizarding families, as well as a list of some of the known half-bloods. Note that some pure-bloods have mixed with muggles, and have become half-bloods.

You must decide what to do when a name occurs in both lists - either respect the tradition of the pure-blood, and ignore the half-blood part, or take the stricter approach and mark any possible half-blood as just that, half-blood.

Inquisitorial Squad

The user must be able to appoint students to the inquisitorial squad, and remove them again. Any number of students can be appointed, but only pure-blood or students from Slytherin (should any non-pure-blood be in that house).

Hacking

You must create a function called `hackTheSystem()` that performs, or activates, the hacking when called. You can add a secret button or keystroke that the user can enter to hack the system, but it must also be possible to do from the console.

Please document in your report, how the system can be hacked.

Hacking must result in (atleast) three things happening:

1. You will be injected, with your own name, into the list of students. You should remain in the lists, until a reload happens, and if the user tries to expel you, there should be a warning of some sorts - you cannot be expelled!
2. The blood-status is no longer thrustworthy. Former pure-bloods will get completely random blood-status, whereas half- and muggle-bloods will be listed as pure-blood. If you can randomly modify the former pure-bloods on every redisplay (sort or filter) of the list, the better!
3. Adding a student to the inquisitorial squad will only work for a limited time, before the student is automatically removed again. Preferably so the user notices that the student gets removed.

Calling `hackTheSystem()` multiple times, shouldn't make any difference - so make sure that the system keeps track of whether it has been hacked.

The effects of the hacking must NOT be happening before `hackTheSystem()` has been called! This means that the blood-status must be correct, until the function is called!

Visual design

You are free to design the web-page as you like, with the few requirements about images, crests and housecolors.

However, make it as visually interessting an experience as possibly. Don't just have an ul of student-names with buttons, but use your design-skills to make something that looks cool!

Add animations and visual feedback. When expelling a student, don't just reload the list without that student, but add an animation that removes the student visually from the list.

When preventing the user from expelling you, be as creative as possible. Think about bad hacker-movies from the 90s! Go crazy, use video and sounds if needed!

Give visual feedback when revoking inquisitorial squad membership from the timer. Make sure that the user sees that the status has changed! You want the hacking to be obvious.

Think about what to include in the list visually. Prefect-, blood, inquisitorial-squad status? Could you use icons? Make it visually exiting!

It isn't a requirement that you draw everything yourself, but be aware that you have the rights for the graphics you use on the page. Check the license on the graphics you find. If you need to credit the creator, do so in a footer on the page, as well as in your report.

Code design

You must use pure vanilla JavaScript (with CSS and HTML) for your solution. No frameworks or plugins.

Your code must run in "strict" mode, and you must use ES6 JavaScript. That means let and const rather than var, and object literals, rather than .prototype definitions.

Write all the code yourself. If you need to use a part you have found on the web, then black-box it as much as possible, and remember to identify where you got it. That means put it in a function, or have comments before and after the lines. Put the URL in the comment, with a short description of what the code is supposed to do.

Use as many functions as possible. Many smaller functions rather than a few large. Think separation of concerns, as well as split between model, view and controller.

The code must use at least one object, a Student-object, with all the properties of a student. Create your own object, when reading the JSON-file, and calculate the properties from that file. Also let some properties have default values, if not found in the file.

You aren't required to add methods or setters and getters to the object, but are welcome to try!

Process and code design

It is recommended that you split the project into several smaller "sprints":

1. Displaying the list of students, with sorting, filtering, searching and details popup.
2. Expelling and prefects
3. Blood-status and inquisitorial squad
4. Hacking

For each sprint, draw an activity diagram of the major activities - you don't need to redraw the sorting and filtering-diagram for the remaining sprints.

Make sure you are completely clear about the flow of activities for expelling and prefects - maybe try to implement expelling before starting the design on prefects.

Get someone else to "play out" your activity diagrams - preferably someone you don't discuss your daily work with.

Documentation

The assignment must be handed in as a report with this information:

- Wireframe for the visual design - include all the dialog-boxes and modals!
- An overview of the various components of the application
- A list of features / user stories (e.g.: sort by first name, sort by last name, etc.)
- A can/can't list of each component or activity, with a note on whether you know how to do it or not (at the beginning of the design)
- Pseudocode or activity diagrams for algorithms:
    - Expelling (incl check for not-expelling you)
    - Prefects
    - Blood-status (original decision-algorithm)
    - hackTheSystem function and additional hacking-features
- Complete activity diagram for all the functions, which parameters they receive, and values they return
- A list of Student-object properties, e.g. firstName, lastName, imageName and so on. Could be a screengrab of the Student object

Collect all documentation in one pdf, make sure it is readable on screen as well as (theoretically) print-out.

The frontpage must include

- Your name
- A screenshot of the product
- A link to the final solution (written out, as well as clickable)
- A link to your github repository (written out, as well as clickable)

Hand-in details

Important: This is a **mandatory submission**, which means that you **must** hand it in, to be eligble for the exam. If you miss the deadline, you will get a new wiseflow-invite before the start of the exam-project, where you will have to re-hand-in.

The assignment is intended to be individual, but you are encouraged to work together, and help each other out. You can also pair up, and solve the assignment together – as long as you don’t divide it into coding and designing, or worse! If you pair up, ensure that the hacking inserts both of you into the list of students!

Where

Hand-in to **Wiseflow**, in the flow: Hacked Hogwarts

How

A pdf as described under documentation.

The pdf must have a link to your working solution, as well as to your github repository

The links must be written as their full urls – as well as be clickable

When

Before the **Wiseflow** deadline (The same as this assignment)

Naming

Name the pdf: *your-name*_hacked_hogwarts.pdf

Where *your-name* is replaced by your actual name. If working as a pair, both of your names!

Feedback

> The assignment will be given an approved / not-approved in wiseflow.
>
> If not-approved, you will have to re-submit before the exam. You will get a new invite for Wiseflow should that be the case.