

## Project Features

1. Create weekly schedule suggestions (functional)
  - a. The project should output a schedule for the user. If the given classes are not available, print the message.
2. Filter classes (functional)
  - a. Users should be able to filter available classes based on given criteria
3. Search classes (functional)
  - a. Users could search for the information of classes with the project.
4. Easily import data (functional)
  - a. The school (or users) should be able to input data, likely in a csv format
5. Easily export data (pdf, etc.) (functional)
  - a. The project could output a file that stores the result. Maybe a pdf file of the schedule, or maybe a file that could link to Google.
6. Optimize schedule choices based on user preferences (functional)
  - a. The program should take the user's limitations on classes and spit out suggested schedules that allow for a prompt graduation and fit within their guidelines

## Project requirements

- 1) Create a SQL database capable of storing all class data in third normal form (nonfunctional) (Feb 21)
  - a) We need to create a back end for our scheduler
- 2) Scrape class data and insert into database using BeautifulSoup (nonfunctional) (Feb 21)
  - a) In order to get the data from course catalogues we will be using BeautifulSoup so that we can obtain the data that we need.
- 3) Create frontend that accesses database with no noticeable lag (nonfunctional)(Feb 28)
  - a) We will be creating the website using HTML and CSS so that the user gets a nice view of the front page. We need to be able to fetch the data from the backend so that we can get the information we need to build a schedule.
- 4) Use Node.js to facilitate lagless communication between frontend and backend (March 13th) (nonfunctional)
  - a) We need to be able to efficiently store our data in the backend and still bring it to the user in a timely fashion, so we use Node.js to communicate between our SQL backend and our HTML/CSS frontend
- 5) Add Filter/search classes feature (functional)(March 20th)
  - a) The user could have a schedule that has specific requirements.
- 6) Create schedule suggestions based on criteria (functional)(March 20th)
  - a) We're going to use algorithmic matching to give students schedules that match all of their criteria
- 7) Add exportation of data (functional) (April 3th)
  - a) Output a file that stores the result.

## Project Plan: Trello

The screenshot shows a Trello board for a project named "Segfaults". The board is organized into four columns: "To Do", "Up next", "In progress", and "Done".

- To Do:**
  - Card: "Create Frontend for accessing Backend" (Due: Feb 28, Progress: 0/2)
  - Card: "Use Node.js to facilitate communication between frontend and backend" (Due: Mar 13, Progress: 0/2)
  - Card: "Create schedule suggestions based on criteria" (Due: Mar 20)
  - Card: "Add filter/class search feature" (Due: Mar 20)
  - Card: "Add exportation of data" (Due: Apr 3)
  - + Add another card
- Up next:**
  - Card: "Create SQL Database" (Due: Feb 21, Progress: 0/4)
  - Card: "Scrape Class Data & Insert into Database" (Due: Feb 21, Progress: 0/4)
  - + Add another card
- In progress:**
  - + Add a card
- Done:**
  - + Add a card

The top navigation bar includes the board name "Segfaults", a star icon, a "Free" label, a "Private" lock icon, a list of members (MB, AP, PY, YM), an "Invite" button, a "Butler" icon, and a "Show Menu" button.