Team Name: Teenage Mutant FullStack Turtles

College Application Tracker

By: James Castro, Joseph Spivack, Nathaniel Chan, Deanna Liu

CSE 416

Homework #9: Code 3

# 

# Table of Contents

[**Table of Contents**](#_ixkjzm4g4h16) **2**

[**Implementation Status Report**](#_ylpfk7izvrwx) **2**

[**Table Creation Statements**](#_i5hvormszbak) **3**

[**Installation Manual**](#_6c10jez8118f) **5**

[**Test Report**](#_at899y32xe6f) **6**

[**Contributions**](#_yzwt9tk309nm) **6**

[James Castro](#_2o2wm4yjhme) 6

[Joseph “Joey” Spivack](#_b8yv65lz6oec) 6

[Nathaniel “Nate” Chan](#_67aa72juuw4d) 6

[Deanna Liu](#_vrt8y6jyd9di) 6

[**Code**](#_5ghu3j90tlpa) **7**

[**Requirements**](#_i0dt714pylw) **7**

[**Design**](#_6ypdvth0j50i) **7**

[Web Server](#_9qn9ldexd7yi) 7

[Web Browser](#_f5vq5w1e8iw8) 7

[WebScraper](#_y7fs9hllji24) 7

[Student and School Database](#_6ghgfs1wqkzt) 7

[User Interface (UI)](#_vly8lo4xfqxt) 7

[Student UI](#_qy0bcu2b7lgu) 7

[Administrator UI](#_66s38d3z8n8q) 8

### **Implementation Status Report**

Please refer to “StatusReport.xlsx” within the doc directory.

### **Table Creation Statements**

CREATE TABLE User (

id INT AUTO\_INCREMENT PRIMARY KEY,

user\_name VARCHAR(50) NOT NULL UNIQUE,

password CHAR(64) NOT NULL,

type ENUM ('Student', 'Admin') NOT NULL,

first\_name VARCHAR(50),

last\_name VARCHAR(50),

email VARCHAR(255)

);

CREATE TABLE Admin (

id INT,

PRIMARY KEY(id),

FOREIGN KEY(id) REFERENCES User(id)

);

CREATE TABLE HighSchool (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) NOT NULL,

city VARCHAR(255) NOT NULL,

state CHAR(2),

ap\_enroll DOUBLE,

sat\_score INT,

act\_score INT,

interested\_majors VARCHAR(255),

interested\_schools VARCHAR(500)

);

CREATE TABLE Student (

id INT,

PRIMARY KEY(id),

FOREIGN KEY(id) REFERENCES User(id) ON DELETE CASCADE,

hs\_id INT,

FOREIGN KEY(hs\_id) REFERENCES HighSchool(id),

financial\_status INT,

major1 VARCHAR(255),

major2 VARCHAR(255),

grad\_year INT,

sat\_math INT,

sat\_ebrw INT,

act\_eng INT,

act\_math INT,

act\_reading INT,

act\_science INT,

act\_comp INT,

sat\_lit INT,

sat\_us INT,

sat\_mathI INT,

sat\_mathII INT,

sat\_eco INT,

sat\_mol INT,

sat\_chem INT,

sat\_phy INT,

numAPs INT,

gpa FLOAT

);

CREATE TABLE School (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(255) UNIQUE NOT NULL,

city VARCHAR(255),

state CHAR(2),

region ENUM ('Northeast', 'Midwest', 'South', 'West'),

admission\_rate DOUBLE,

cost DOUBLE,

ranking INT,

size INT,

act\_composite INT,

sat\_math INT,

sat\_ebrw INT,

act\_range\_low INT,

act\_range\_high INT,

sat\_math\_range\_low INT,

sat\_math\_range\_high INT,

sat\_ebrw\_range\_low INT,

sat\_ebrw\_range\_high INT,

avg\_accepted\_gpa FLOAT,

accepted\_gpa\_high FLOAT,

accepted\_gpa\_low FLOAT

);

CREATE TABLE Majors (

school\_id INT,

major VARCHAR(255),

PRIMARY KEY(school\_id, major),

FOREIGN KEY(school\_id) REFERENCES School(id)

);

CREATE TABLE Applications (

student\_id INT,

college\_id INT,

FOREIGN KEY(student\_id) REFERENCES Student(id) ON DELETE CASCADE,

FOREIGN KEY(college\_id) REFERENCES School(id) ON DELETE CASCADE,

PRIMARY KEY(student\_id, college\_id),

questionable BOOL,

status ENUM('Pending', 'Waitlisted', 'Accepted', 'Rejected', 'Deferred', 'Withdrawn')

);

CREATE TABLE LoggedIn (

id INT,PRIMARY KEY(id),

FOREIGN KEY(id)

REFERENCES User(id) ON DELETE CASCADE,

random\_val INT

);

### **Installation Manual**

Setup:

Step 1) Download and setup a mySQL community server

[https://dev.mysql.com/downloads/]

Step 2) Create a database in the mysql monitor and call the create table statements as

specified in the above in the persistence section of this document.

Step 3) Assuming that you haven't cloned the CSE416Project repo or extracted the zip

file.

Step 4) Change the values in the config.json to the correct information related to your

database setup, all values in the config in the backend folder starting with

‘db\_’ relate to the database and may need to be changed. If you are installing

my sql locally the host will be localhost and you can use root as your account.

You may need to change the security type for the root account. Using the

mySQL statement:

ALTER USER ‘username’@’host’ IDENTIFIED WITH mysql\_native\_password

BY ‘new password’

Setting that up locally it looks like:

ALTER USER ‘root’@’localhost’ IDENTIFIED WITH mysql\_native\_password

BY ‘new password’

Step 5) Make sure you have node and npm installed. Installing is simple and easy!

First, make sure you have [https://nodejs.org/en/download/] (nodeJS)

Installed. If you have node installed properly, you also have npm! Npm used to

mean node package manager. The project has a variety of dependencies that

npm can install quite easily. Simply cd into the respective directories (c4me)

and type "npm install".

Step 6) In the backend directory of the CSE416 repo also call npm install.

Step 7) To start the server start the app.js file by calling “node app.js” in the backend

directory. Nodemon is also useful for development as it will auto restart the

server whenever it detects changes to the files. To use nodemon make sure it's

installed (using npm) and run the app.js with nodemon rather than node.

Step 8) Confirm that your mysql server is running on your machine. On linux this can

be done by calling “service mysql status”

Step 9) In the c4me directory, call npm start.

### **Test Report**

Please refer to “TestReport.pdf” within the doc directory for the testing done during

this checkpoint of c4me.

### **Contributions**

#### James Castro

* Main activities and work products:
  1. Connecting frontend (UI) to the backend (servers).
  2. Converted html/css documentation to React.
  3. Assisted in implementing the backend servers to function properly.

#### Joseph “Joey” Spivack

* Main activities and work products:
  1. Creating & implementing the SQL statements for the servers.
  2. Connecting frontend (UI) to the backend (servers).
  3. Implementing the backend servers to function properly.

#### Nathaniel “Nate” Chan

* Main activities and work products:
  1. Shared ideas for designs for frontend design.
  2. Implementing the backend servers to function properly.
  3. Testing backend and fullstack

#### Deanna Liu

* Main activities and work products:

1. Organizing documentation.
2. Shared ideas for designs for frontend design.
3. Converted html/css documentation to React.
4. Completed the template and ideas for the frontend design.

### **Code**

Please refer to the “src” folder within this (zip) folder for the code.

### **Requirements**

Please refer to “Requirements.pdf” within this directory.

### **Design**

Please refer to “Design.pdf” within this directory.

#### Web Server

Node.js server that deals with all front end requests and database interactions. It

serves requests from the browser by using both the web scraping component and database using mySQL. It will also read files directly on the web server.

#### Web Browser

The common browsers (Chromium-based browsers & Firefox) will be used to access the web application. This is done by performing get requests to the React server (Labeled UI) for each website. It will then (according to the site's design) perform get or post requests to the Node.js Web Server to obtain specific data or modify data.

#### WebScraper

College rankings will be obtained from the college ranking site by accessing the site

using Puppeteer. It will take requests to perform the web scraping from the Web

Server and save the results to the Database.

#### Student and School Database

Uses mySQL to store data from the student information Web Server component and

school information web the WebScraper component.

#### User Interface (UI)

Website will be designed using a React server that uses Materialize.css and Bootstrap

as the frontend frameworks. The User’s web browser uses get requests to obtain html

files for each webpage.

#### Student UI

The Student UI is the part of the website only accessible to Students. Authentication

using cookies containing session IDs issued on login will be performed on each sensitive

request to confirm that the user performing the action is allowed to.

#### Administrator UI

The Administrator UI is the part of the website only accessible to Administrators.

Authentication using cookies containing session IDs issued on login will be performed

on each sensitive request to confirm that the user performing the action is allowed to.