DATABASE TABLES SCHEMA

NOTES:

- **The naming convention is camelCase.**
- **All purple variables are countable (substitutes for arrays); front-end should expect functions to retrieve count and actual list.**
- **Blue boxes are notes for the front-end team; red for the back-end team.**
- **Yellow boxes are character limits / hard-coded formats / other restrictions that both teams should be aware of.**
- **The first field is the unique identifier for that item.**
- **Dotted arrows are the same variable for one instance of the item.**

APPLICATIONS TABLE

varchar(50)	varchar(50)	varchar(50)	varchar(10)			varchar(200)	varchar(100)
username (String) (unique)	**firstName** (String)	**lastName** (String)	**year** (String) (1 of 4 options)	**gpa** (float)	**grade320** (char)	**reference** (String)	**attachment** (String) (S3 link)
			'Freshman' etc.			'firstName lastName, email'	

minimum 1, maximum 2, enforced in front-end

generated by front-end operations before sending to database

STUDENTS TABLE

varchar(50)				
username (String) (unique)	**conflict** (int = section ID)	**ranking** (int = 1, 2, ...)	**sectionRanked** (int = section ID)	**matched** (int = section ID)

So, if a student prefers section 7 and then section 8, the pairings would be (1, 7), (2, 8).

Send out notification when this value changes from -1 to section ID, since the student is now matched.

USERS TABLE

varchar(50)	varchar(255)		varchar(50)	varchar(50)
username (String) (unique)	**password** (String)	**type** (Boolean; 0 = student, 1 = professor)	**firstName** (String)	**lastName** (String)

Name will be asked for during registration. This name can be retrieved from here to greet user.

PROFESSORS TABLE

varchar(50)			varchar(50)
username (String) (unique)	**section** (int = section ID)	**ranking** (int = 1, 2, ...)	**appRanked** (String = student's username)

MATCHINGS TABLE

	varchar(50)
section (int = section ID)	**username** (String = student's username)

SECTIONS TABLE

	varchar(10)	varchar(30)	varchar(50)	varchar(50)		
id (integer) (unique, taken from schedule)	**day** (String)	**time** (String)	**username** (String = Professor's username)	**applicant** (String = student's username)	**capacity** (int)	**numEnrolled** (int)
	'MW', 'MWF', 'TuTh', etc.	e.g. '2:30 PM - 3:45 PM'				

Retrieve capacity and numEnrolled from here.

Update numEnrolled as the matching algorithm progresses.

Used to be String = Professor's name. Simplifies back-end operations. Front-end will have function to retrieve instructor's name when displaying section info.

Every student who has submitted application choosing this section. Update with every submission.