James O’Boyle, Marvin Claude, Nicholas Senatore

Professor Krueger

Database Management – CSC 351-D01

November 20th, 2018

**Radio 1851 Database**

The goal of our final project for Professor Krueger’s Database Management class was a creation of a database for a radio station (highly similar to the local radio station here at Saint Joseph’s University, which is what we modeled it after). Considering it is quite a hassle to manually insert every detail every time new DJs come aboard, a simplification of this process is in high demand. As a result, we are capable of providing a scheduling software that the radio station can use to connect its managers, DJs, listeners, and advertisers through one, easy-to-use interface. We anticipate the application to be slowly implemented into the currently existing system at Radio 1851.

The design of this database was driven primarily by the concept that different types of users (managers, DJs, listeners, advertisers) would each have specific privileges while still interacting with one another through the database. Listeners would be able to request songs to specific shows, as well as view the show schedule and countless DJs (and their respective public information). DJs may view song requests for their shows, provide information about themselves, and submit requests for new shows that would later be approved by a given manager. Managers can mentor DJs, add other managers and DJs into the system, delete DJs and their respective shows, and approve either show requests or ads by registered advertisers.

|  |  |  |  |
| --- | --- | --- | --- |
| Manager Homepage | DJ Homepage | Listener Homepage | Advertiser Homepage |
|  |  |  |  |

Image of ERD Diagram:

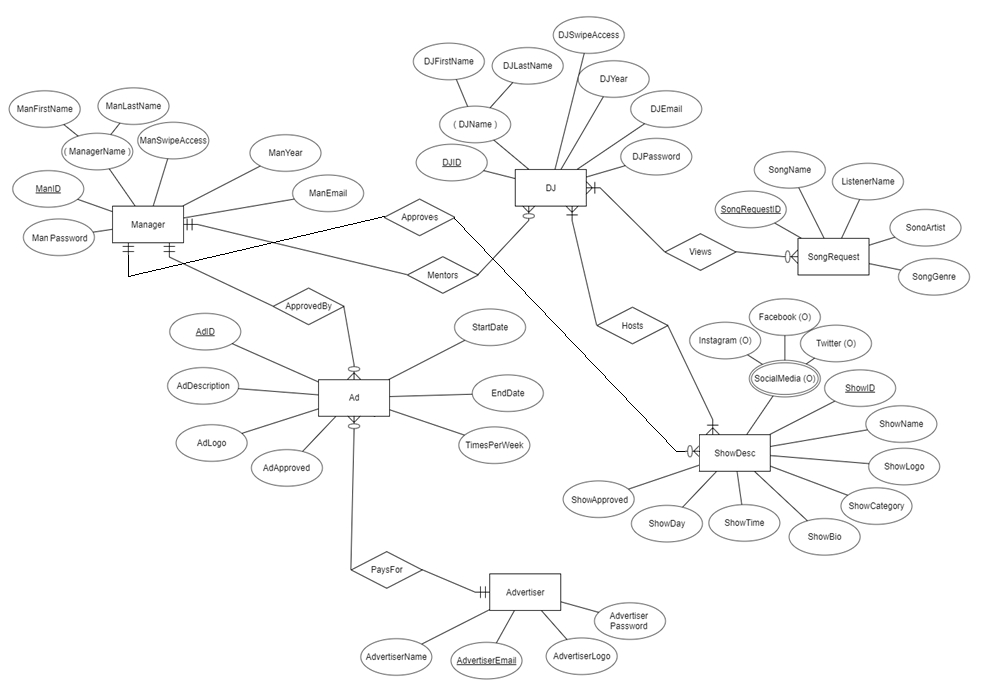
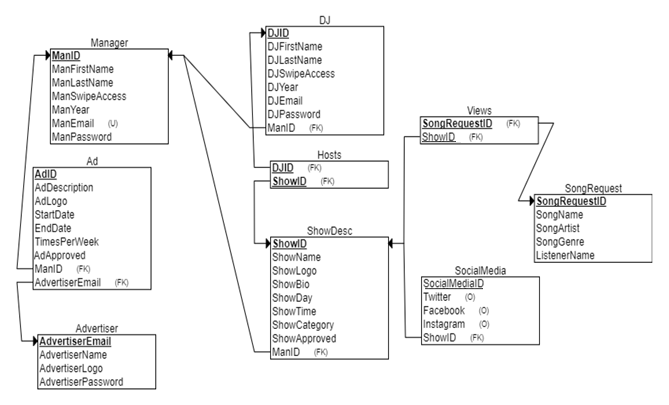


Image of Relational Schema:



SQL for Creation of Tables:

CREATE TABLE Manager ( ManID INT NOT NULL, ManFirstName VARCHAR(255) NOT NULL, ManLastName VARCHAR(255) NOT NULL, ManSwipeAccess CHAR(1) NOT NULL, ManYear INT NOT NULL, ManEmail VARCHAR(255) NOT NULL, ManPassword VARCHAR(255) NOT NULL, PRIMARY KEY (ManID), UNIQUE (ManEmail) );

CREATE TABLE DJ ( DJID INT NOT NULL, DJFirstName VARCHAR(255) NOT NULL, DJLastName VARCHAR(255) NOT NULL, DJSwipeAccess CHAR(1) NOT NULL, DJYear INT NOT NULL, DJEmail VARCHAR(255) NOT NULL, DJPassword VARCHAR(255) NOT NULL, ManID INT NOT NULL, PRIMARY KEY (DJID), FOREIGN KEY (ManID) REFERENCES Manager(ManID), UNIQUE (DJEmail) );

CREATE TABLE Advertiser ( AdvertiserEmail VARCHAR(255) NOT NULL, AdvertiserPassword VARCHAR(255) NOT NULL, AdvertiserName VARCHAR(255) NOT NULL, AdvertiserLogo VARCHAR(255) NOT NULL, PRIMARY KEY (AdvertiserEmail) );

CREATE TABLE ShowDesc ( ShowID INT NOT NULL AUTO\_INCREMENT, ShowName VARCHAR(255) NOT NULL, ShowLogo VARCHAR(255) NOT NULL, ShowBio VARCHAR(255) NOT NULL, ShowCategory VARCHAR(255) NOT NULL, ShowDay VARCHAR(255) NOT NULL, ShowTime VARCHAR(255) NOT NULL, ManID INT NOT NULL, ShowApproved CHAR(1) NOT NULL, PRIMARY KEY (ShowID), FOREIGN KEY (ManID) REFERENCES Manager(ManID) );

CREATE TABLE SocialMedia ( SocialMediaID INT NOT NULL AUTO\_INCREMENT, Facebook VARCHAR(255), Instagram VARCHAR(255), Twitter VARCHAR(255), ShowID INT NOT NULL, PRIMARY KEY (SocialMediaID, ShowID), FOREIGN KEY (ShowID) REFERENCES ShowDesc(ShowID) );

CREATE TABLE SongRequest ( SongRequestID INT NOT NULL AUTO\_INCREMENT, ListenerName VARCHAR(255) NULL, SongName VARCHAR(255) NOT NULL, SongArtist VARCHAR(255) NOT NULL, SongGenre VARCHAR(255) NOT NULL, PRIMARY KEY (SongRequestID) );

CREATE TABLE Views ( ShowID INT NOT NULL, SongRequestID INT NOT NULL, PRIMARY KEY (ShowID, SongRequestID), FOREIGN KEY (ShowID) REFERENCES ShowDesc(ShowID), FOREIGN KEY (SongRequestID) REFERENCES SongRequest(SongRequestID) );

CREATE TABLE Hosts ( DJID INT NOT NULL, ShowID INT NOT NULL, PRIMARY KEY (DJID, ShowID), FOREIGN KEY (DJID) REFERENCES DJ(DJID), FOREIGN KEY (ShowID) REFERENCES ShowDesc(ShowID) );

CREATE TABLE Ad ( AdID INT NOT NULL AUTO\_INCREMENT, AdDescription VARCHAR(255) NOT NULL, AdLogo VARCHAR(255) NOT NULL, StartDate DATE NOT NULL, EndDate DATE NOT NULL, TimesPerWeek INT NOT NULL, AdvertiserEmail VARCHAR(255) NOT NULL, ManID INT NOT NULL, AdApproved CHAR(1) NOT NULL, PRIMARY KEY (AdID), FOREIGN KEY (AdvertiserEmail) REFERENCES Advertiser(AdvertiserEmail), FOREIGN KEY (ManID) REFERENCES Manager(ManID) );

SQL for Population of Tables:

INSERT INTO `manager` (`ManID`, `ManFirstName`, `ManLastName`, `ManSwipeAccess`, `ManYear`, `ManEmail`, `ManPassword`) VALUES

('10663859', 'James', 'OBoyle', 'Y', '2020', 'jo663859@sju.edu', 'james'),

('106633833', 'John', 'doe', 'Y', '2018', 'john@gmail.com', 'john'),

('10373564', 'Oliver', 'Johnson', 'Y', '2020', 'oliver@gmail.com', 'oliver'),

('10987465', 'Jack', 'richards', 'Y', '2017', 'jack@hotmail.com', 'jack'),

('19836456', 'Thomas', 'Knox', 'Y', '2022', 'thomas@aol.com', 'thomas'),

('38747444', 'Henry', 'Seer', 'Y', '2021', 'henry@xboxlive.com', 'henry'),

('12343568', 'William', 'James', 'Y', '2019', 'William@gmail.com', 'james'),

('27883946', 'bob', 'parks', 'Y', '2019', 'bob@gmail.com', 'bob'),

('10298444', 'Gabriel', 'Mejia', 'Y', '2020', 'gabe@hotmail.com', 'gabriel'),

('19288745', 'Vincent', 'Cardenas', 'Y', '2018', 'vincent@yahoo.com', 'vincent'),

('10836456', 'Scott', 'Peterson', 'Y', '2019', 'Scott@yahoo.com', 'scott'),

('10998456', 'Sonny', 'Higgins', 'Y', '2018', 'Sonny@yahoo.com', 'sonny'),

('12009456', 'Ty', 'Frazier', 'Y', '2021', 'ty@sju.edu', 'ty'),

('10983456', 'Devin', 'Clay', 'Y', '2022', 'devin@aim.com', 'devin'),

('10874567', 'Jodi', 'Chan', 'Y', '2018', 'jodi@sju.edu', 'jodi'),

('10663456', 'Branden', 'McPherson', 'Y', '2021', 'Brans@yahoo.com', 'branden'),

('10292836', 'Alexis', 'Cash', 'N', '2022', 'cash@gmail.com', 'alexis'),

('10874534', 'Jim', 'Cho', 'Y', '2018', 'jim@sju.edu', 'jim'),

('10663345', 'Brandon', 'Mcdowell', 'Y', '2021', 'Brandun@yahoo.com', 'brandon'),

('10292987', 'Alex', 'ryan', 'N', '2022', 'alex@gmail.com', 'alex');

INSERT INTO `advertiser` (`AdvertiserEmail`, `AdvertiserPassword`, `AdvertiserName`, `AdvertiserLogo`) VALUES

('slechelle0@feedburner.com', 'adpass1', 'Evonik Stockhausen, LLC', '/uploads/advertiser1.png'),

('eacland1@yahoo.com\r\n', 'adpass2', 'Cantrell Drug Company', '/uploads/advertiser2.png'),

('dcella2@networkadvertising.org', 'adpass3', 'Dispensing Solutions, Inc.', '/uploads/advertiser3.png'),

('lbrackstone3@eepurl.com', 'adpass4', 'Medtronic Neuromodulation', '/uploads/advertiser4.png'),

('smarron4@blogtalkradio.com', 'adpass5', 'Watson Pharma, Inc.', '/uploads/advertiser5.png'),

('sdawby5@freewebs.com\r\n', 'adpass6', 'Zenith Medicosm SL', '/uploads/advertiser6.png'),

('wstannah6@foxnews.com', 'adpass7', 'Nelco Laboratories, Inc.', '/uploads/advertiser7.png'),

('tmilliken7@whitehouse.gov', 'adpass8', 'Deb USA, Inc.', '/uploads/advertiser8.png'),

('plaugheran8@auda.org.au', 'adpass9', 'GE Healthcare Inc.', '/uploads/advertiser9.png'),

('ainchcomba@webs.com', 'adpass10', 'King Bio Inc.', '/uploads/advertiser10.png'),

('lmaryott9@unblog.fr', 'adpass11', 'Laboratoires Clarins S.A.', '/uploads/advertiser11.png'),

('rdellascalab@barnesandnoble.com', 'adpass12', 'Nelco Laboratories, Inc.', '/uploads/advertiser12.png'),

('denrricoc@hostgator.com', 'adpass13', 'CVS Pharmacy', '/uploads/advertiser13.png'),

('kgyenesd@reuters.com', 'adpass14', 'TOO FACED COSMETICS\r\n', '/uploads/advertiser14.png'),

('htramele@wikimedia.org', 'adpass15', 'Scarguard Labs, LLC', '/uploads/advertiser15.png'),

('ssollisf@hibu.com', 'adpass16', 'Amerisource Bergen', '/uploads/advertiser16.png'),

('sgaitskellg@china.com.cn', 'adpass17', 'Deliz Pharmaceutical Corp', '/uploads/advertiser17.png'),

('nscotfurthh@comsenz.com', 'adpass18', 'Sanum Kehlbeck GmbH & Co. KG', '/uploads/advertiser18.png'),

('jozeltonk@wisc.edu', 'adpass19', 'Walgreen Company', '/uploads/advertiser19.png'),

('lmeasorl@deliciousdays.com', 'adpass20', 'Medtech Products Inc.', '/uploads/advertiser20.png');

INSERT INTO `ad` (`AdID`, `AdDescription`, `AdLogo`, `StartDate`, `EndDate`, `TimesPerWeek`, `AdvertiserEmail`, `ManID`, `AdApproved`) VALUES

(1, 'this ad is about tires', '/uploads/advertisement01.png', '2018-11-26', '2018-12-06', '3', 'ainchcomba@webs.com', '10836456', 'Y'),

(2, 'this ad is about ice cream', '/uploads/advertisement02.png', '2018-11-26', '2018-12-26', '3', 'dcella2@networkadvertising.org', '106633833', 'Y'),

(3, 'this ad is about candy', '/uploads/advertisement3.png', '2018-11-27', '2019-01-23', '1', 'eacland1@yahoo.com\r\n', '10292987', 'N'),

(4, 'this ad is about bananas', '/uploads/advertisement4.png', '2018-11-12', '2018-11-30', '4', 'lmeasorl@deliciousdays.com', '10373564', 'N'),

(5, 'this ad is about music', '/uploads/advertisement5.png', '2018-12-02', '2018-12-09', '5', 'nscotfurthh@comsenz.com', '38747444', 'Y'),

(6, 'this ad is about public safety', '/uploads/advertisement6.png', '2018-11-27', '2018-11-30', '2', 'rdellascalab@barnesandnoble.com', '10292836', 'Y'),

(7, 'this ad is about St joes', '/uploads/advertisement7.png', '2018-11-28', '2019-01-24', '4', 'tmilliken7@whitehouse.gov', '38747444', 'Y'),

(8, 'this ad is about matchbox cars', '/uploads/advertisement8.png', '2018-11-27', '2019-02-13', '3', 'sgaitskellg@china.com.cn', '12343568', 'Y'),

(9, 'this ad is about chia pets', '/uploads/advertisement9.png', '2018-11-28', '2019-03-07', '1', 'nscotfurthh@comsenz.com', '10874534', 'Y'),

(10, 'this ad is about bricks', '/uploads/advertisement10.png', '2018-11-20', '2019-03-14', '4', 'wstannah6@foxnews.com', '10983456', 'Y'),

(11, 'this ad is about headphones', '/uploads/advertisement11.png', '2018-11-14', '2019-02-14', '2', 'kgyenesd@reuters.com', '27883946', 'Y'),

(12, 'this ad is about adopting children', '/uploads/advertisement12.png', '2018-11-20', '2019-02-14', '2', 'sgaitskellg@china.com.cn', '12343568', 'Y'),

(13, 'This ad is one of those really sad dog adds', '/uploads/advertisement13.png', '2018-11-29', '2019-02-20', '2', 'eacland1@yahoo.com\r\n', '10298444', 'N'),

(14, 'this ad is about cats', '/uploads/advertisement14.png', '2018-11-27', '2019-03-27', '2', 'lbrackstone3@eepurl.com', '10987465', 'N'),

(15, 'this ad is about Basketball', '/uploads/advertisement15.png', '2018-11-27', '2018-11-29', '5', 'smarron4@blogtalkradio.com', '12009456', 'Y'),

(16, 'This ad is about lincoln logs', '/uploads/advertisement16.png', '2018-11-21', '2019-02-19', '2', 'eacland1@yahoo.com\r\n', '10663345', 'Y'),

(17, 'this ad is a public service announcement', '/uploads/advertisement17.png', '2018-11-27', '2019-06-06', '1', 'tmilliken7@whitehouse.gov', '10663456', 'Y'),

(18, 'this ad is about septa', '/uploads/advertisement18.png', '2018-11-15', '2019-04-25', '2', 'eacland1@yahoo.com\r\n', '10663859', 'N'),

(19, 'This ad is about the radio stations shows', '/uploads/advertisement19.png', '2018-11-30', '2019-04-24', '2', 'kgyenesd@reuters.com', '10663456', 'N'),

(20, 'This ad is about nursing homes', '/uploads/advertisement20.png', '2018-11-23', '2019-05-16', '4', 'lmeasorl@deliciousdays.com', '10998456', 'N');

INSERT INTO `dj` (`DJID`, `DJFirstName`, `DJLastName`, `DJSwipeAccess`, `DJYear`, `DJEmail`, `DJPassword`, `ManID`) VALUES

('1', 'India', 'Hendrix', 'Y', '2019', 'wfroude1@cnbc.com', 'india', '10874567'),

('2', 'Alysia', 'Smart', 'Y', '2020', 'coldall7@addtoany.com', 'alysia', '10292836'),

('3', 'Arham', 'Zamora', 'Y', '2019', 'wdagg8@gravatar.com', 'arham', '10373564'),

('4', 'Helena', 'Castro', 'Y', '2019', 'soaker9@craigslist.org\r\n', 'helena', '10663859'),

('5', 'Eva', 'Paul', 'Y', '2022', 'dtomkowiczc@facebook.com', 'eva', '10292987'),

('6', 'James', 'May', 'Y', '2021', 'jyouensd@mit.edu', 'james', '10663456'),

('7', 'Callum', 'Hook', 'Y', '2022', 'kdumbeltonf@odnoklassniki.ru', 'callum', '12009456'),

('8', 'Charity', 'Clifford', 'N', '2021', 'dmatthiusg@prlog.org', 'charity', '10983456'),

('9', 'Nick', 'Brock', 'Y', '2022', 'jdexterh@ezinearticles.com', 'anand', '10298444'),

('10', 'Marvin', 'Campbell', 'Y', '2020', 'nmacgianyn@godaddy.com', 'bruce', '38747444'),

('11', 'Julie', 'Ruiz', 'Y', '2022', 'mmorrids@macromedia.com', 'julie', '106633833'),

('12', 'Gino', 'Keller', 'Y', '2022', 'pcorneliussenu@tripod.com', 'gino', '12343568'),

('13', 'Rudy', 'Britt', 'Y', '2022', 'cdomanx@earthlink.net', 'rudy', '10663456'),

('14', 'Hassan', 'Noble', 'Y', '2022', 'bmendez17@zdnet.com', 'hassan', '38747444'),

('15', 'Mindy', 'Cruz', 'N', '2019', 'rrubbens1b@bravesites.com', 'mindy', '106633833'),

('16', 'Tanner', 'Dodson', 'Y', '2019', 'speet1m@surveymonkey.com', 'tanner', '10874534'),

('17', 'Darsh', 'Hammond', 'N', '2019', 'hsambeck1j@com.com', 'darsh', '27883946'),

('18', 'Nichola', 'Good', 'Y', '2022', 'gravenhills2m@dagondesign.com', 'nichola', '27883946'),

('19', 'Kylo', 'Scott', 'Y', '2021', 'cchampneys2q@github.com', 'kylo', '12343568'),

('20', 'Jace ', 'Seirra', 'Y', '2018', 'bjossel0@dmoz.org', 'jace', '10663345');

INSERT INTO `songrequest` (`SongRequestID`, `ListenerName`, `SongName`, `SongArtist`, `SongGenre`) VALUES

(1, 'Corey', 'Stairway to Heaven', 'Led Zeppelin', 'Rock'),

(2, 'Bill', 'Holiday', 'Green Day', 'Rock'),

(3, 'Marvin', 'Sicko Mode', 'Travis Scott', 'Rap'),

(4, 'Nick', 'Yellow submarine', 'The Beatles', 'Rock'),

(5, 'John', 'Hail Mary', 'Tupac', 'Rap'),

(6, 'Jack', 'SweatPants', 'Childish Gambino', 'Rap'),

(7, 'Jessica', 'Hey There Delilah', 'Plain White T\'s', 'Alternative'),

(8, 'Anonymous', 'White Iverson', 'Post Malone', 'Rap'),

(9, 'Kristen', 'Atlantic city', 'The Band', 'Rock'),

(10, 'Tommy', 'Uproar', 'lil Wayne', 'Rap'),

(11, 'Kate', 'Brown eyed girl', 'Van Morrison', 'Rock'),

(12, 'Ericka', 'Piano Man', 'Billy Joel', 'Rock'),

(13, 'Patrick', 'T-Shirt', 'Thomas Rhett', 'Country'),

(14, 'Jen', 'Mr.Brightside', 'The Killers', 'Alternative'),

(15, 'Jill', 'Helena beat', 'Foster the People', 'Alternative'),

(16, 'Anonymous', 'elenor rigby', 'the beatles', 'Rock'),

(17, 'Grover', 'Sit next to me', 'Foster the People', 'Alternative'),

(18, 'Lee', 'The spins', 'mac miller', 'Rap'),

(19, 'dillon', 'Juke jam', 'Chance the Rapper', 'Rap'),

(20, 'vince', 'Casey Jones', 'Grateful Dead', 'Rock');

INSERT INTO `showdesc` (`ShowID`, `ShowName`, `ShowLogo`, `ShowBio`, `ShowCategory`, `ShowDay`, `ShowTime`, `ManID`, `ShowApproved`) VALUES

(1, 'rock hour', '/uploads/show1.png', 'This show is about classic rock music', 'Music', 'Monday', '08:00 PM', '10292987', 'N'),

(2, 'best show ever', '/uploads/show2.png', 'This show is a talk show about life', 'Talk show', 'Tuesday', '08:00 PM', '27883946', 'N'),

(3, 'no name show', '/uploads/show3.png', 'This show is nothing at all', 'music', 'Monday', '09:00 PM', '10292836', 'N'),

(4, 'best show of all time', '/uploads/show4.png', 'This show is a talk show about current events', 'Talk show', 'Tuesday', '09:00 PM', '10663345', 'Y'),

(5, '1 and only 1', '/uploads/show5.png', 'this show is a music broadcast', 'Music', 'Wednesday', '10:00 PM', '10998456', 'Y'),

(6, 'james show', '/uploads/show6.png', 'This show is about new york sports', 'Sports', 'Thursday', '10:00 PM', '10836456', 'N'),

(7, 'rock show', '/uploads/show7.png', 'This show plays classic rock ', 'Music', 'Friday', '10:00 PM', '10998456', 'Y'),

(8, 'new music show', '/uploads/show8.png', 'this show plays new music', 'Music', 'Monday', '11:00 PM', '10836456', 'N'),

(9, 'nicks show', '/uploads/show9.png', 'this show is about technology', 'Talk show', 'Tuesday', '11:00 PM', '10663859', 'Y'),

(10, 'marvins show', '/uploads/show10.png', 'this is a show that plays alt hits', 'Music', 'Wednesday', '11:00 PM', '10373564', 'Y'),

(11, 'my show ', '/uploads/show11.png', 'A show where dj talks about his life', 'Talk show', 'Thursday', '11:00 PM', '38747444', 'Y'),

(12, 'sju show', '/uploads/show12.png', 'a show about st. joes', 'Talk show', 'Friday', '11:00 PM', '10874567', 'Y'),

(13, 'hawk talk', '/uploads/show13.png', 'a show about st joes sports', 'Talk show', 'Monday', '01:00 PM', '10874534', 'Y'),

(14, 'philly sports show', '/uploads/show14.png', 'a show about philly sports', 'Sports', 'Tuesday', '01:00 PM', '10663456', 'Y'),

(15, 'sports talk show', '/uploads/show15.png', 'a show about sports in general', 'Sports', 'Wednesday', '01:00 PM', '19288745', 'Y'),

(16, 'life and love show', '/uploads/show16.png', 'a show about life and love problems', 'Talk show', 'Thursday', '01:00 PM', '10874534', 'N'),

(17, 'vent session show', '/uploads/show17.png', 'a show where callers can call in and vent to the dj', 'Talk show', 'Friday', '01:00 PM', '10987465', 'Y'),

(18, 'weird show ', '/uploads/show18.png', 'a show for everything weird and wacky', 'Talk show', 'Saturday', '01:00 PM', '10298444', 'Y'),

(19, 'the show that cant agree on a name', '/uploads/show19.png', 'a show where the two hosts fight about what to play', 'Talk show', 'Sunday', '02:00 PM', '12009456', 'Y'),

(20, 'st joes pros', '/uploads/show20.png', 'a show for upperclassman to teach younger people about st. joes', 'Talk show', 'Monday', '07:00 PM', '106633833', 'N');

INSERT INTO `hosts` (`DJID`, `ShowID`) VALUES

('1', '1'),

('2', '2'),

('3', '3'),

('4', '4'),

('5', '5'),

('6', '6'),

('7', '7'),

('8', '8'),

('9', '9'),

('10', '10'),

('11', '11'),

('12', '12'),

('13', '13'),

('14', '14'),

('15', '15'),

('16', '16'),

('17', '17'),

('18', '18'),

('19', '19'),

('20', '20');

INSERT INTO `views` (`ShowID`, `SongRequestID`) VALUES

('9', '1'),

('9', '2'),

('9', '3'),

('2', '4'),

('2', '5'),

('10', '6'),

('6', '7'),

('9', '8'),

('9', '9'),

('10', '10'),

('8', '11'),

('8', '12'),

('8', '13'),

('9', '14'),

('8', '15'),

('9', '16'),

('6', '17'),

('9', '18'),

('9', '19'),

('9', '20');

INSERT INTO `socialmedia` (`SocialMediaID`, `Facebook`, `Instagram`, `Twitter`, `ShowID`) VALUES

(1, NULL, NULL, NULL, '1'),

(2, 'facebook.com/bestshowever', 'instagram.com/bse', 'twitter.com/bse', '2'),

(3, 'facebook.com/nonameshow', NULL, NULL, '3'),

(4, NULL, 'instagram.com/bsoat', 'twitter.com/bsoat', '4'),

(5, NULL, NULL, 'twitter.com/show1andonly1', '5'),

(6, NULL, 'instagram.com/jamesshow', 'twitter.com/jamesshow', '6'),

(7, NULL, 'instagram.com/rockshow', 'twitter.com/rockshow', '7'),

(8, NULL, 'instagram.com/nms', NULL, '8'),

(9, NULL, 'instagram.com/nicksshow', 'twitter.com/nickshows', '9'),

(10, NULL, 'instagram.com/ms', NULL, '10'),

(11, 'facebook.com/myshow', NULL, NULL, '11'),

(12, 'facebook.com/sjushow', NULL, NULL, '12'),

(13, 'facebook.com/hawktalk', NULL, 'twitter.com/hawktalk', '13'),

(14, NULL, NULL, NULL, '14'),

(15, NULL, 'instagram.com/sts', 'twitter.com/sts', '15'),

(16, 'facebook.com/landl', 'instagram.com/landloveshow', 'twitter.com/landl', '16'),

(17, 'facebook.com/vss', NULL, 'twitter.com/vent', '17'),

(18, NULL, 'instagram.com/weirddddd', 'twitter.com/weird', '18'),

(19, NULL, 'instagram.com/cantagree', 'twitter.com/cantagree', '19'),

(20, 'facebook.com/sjp', NULL, 'twitter.com/sjp', '20');

For example, the code above for populating the “DJ” table in our database does so with 20 different DJs. It includes the DJID (which is auto incremented normally when adding via PHP) along with the DJs first and last name, whether or not they have swipe access, their email, their password to login, and a foreign key of manID, which corresponds to which manager mentors that individual DJ.

When entering data for the values that require a photo, a string that directs the PHP code to the image is inserted. As a result, whoever runs this PHP code must have an “uploads” folder in their HTDOCS, and all of the accompanying .pngs. These PNGs are placed onto our GIT Repository in a folder for all to obtain.

Once the database had been populated, the next step was to ensure that the database was normalized. Such was accomplished by ensuring that each attribute in the table was functionally dependent on the primary key (such as DJID). This would guarantee that the primary key in each table would determine all values in any corresponding rows of a particular column. As a result, for example, the primary key “DJID” determines all of the non-primary key attributes that go hand in hand with it. In this case, such attributes are the DJ’s first and last name, their swipe access status, their email, and their password to login. By making sure that these states of functional dependencies were present in all tables in our database, we were able to confirm that each entity in our database was normalized to the greatest extent.

We implemented constraints to ensure that the type of data that a given user enters into a table will not only be correct, but complete and lacking of any nonsensical values. AdID, AdvertiserEmail, DJID, ManID, ShowID, SocialMediaID, and SongRequestID are all listed as “primary keys” so that each table would have a unique row to help identify it. Foreign keys were also implemented so that rows in other tables may be uniquely identified. The foreign keys used include ManID, AdvertiserEmail, DJEmail, ShowID, SongReqID, etc. Lastly, the “Unique” constraint implemented would ensure that values that could not be the same as others, like DJID, ADID, or even emails, would not have matching data with information that was in the same column.

To ensure referential integrity, each user would be originally faced with a login page. As such, only users who were authorized to make changes to certain tables would have the ability to do so. “NOT NULL” constraints were used in the creation of countless variables in the tables to prevent missing column data. Adding this constraint also ensured that users would not delete foreign key values by accident, thus ruining the tables where the foreign key would be a primary key, making it difficult for such tables in the database as a whole to remain connected.

Error checking was accomplished by not allowing incorrect or nonsensical data to be entered into the database / forms. One way we were able to ensure that users would only input data that our system agreed with was to implement drop down menus on the entry forms, such as “ShowDay” and “ShowTime”. This way we may control both the data entered and their respective formats. We also made sure to make all primary keys and foreign keys “NOT NULL”. This way we could make sure that our users entered all the data necessary in the form. Various if statements and SQL Selects were used to compare values within the database to prevent overlapping or corruption of data, as well. For example, if a DJ tries to add a show to a time that already exists, he or she is unable to do so and is prompted accordingly. If a DJ tries to add a show with a HostID or CohostID that does not correspond to a DJID in the system, their request is denied, and they are told to enter correct information.

Transactions are vital in guaranteeing that our database would be reliable. Transactions ensure that the data in our database will always remain correct, intact, and follow the ACID format. One place where transactions could be used in our database are in the table where DJ can submit show times for certain shows. For example, if two different DJs are attempting to request a show at the same time, this could potentially cause a major problem (similar to individuals attempting to buy the same airplane seat at the same time).

Lastly, to increase efficiency within our database we could use indexes. Three indexes are:

* IDX\_MENTOR\_ID which would create an index on the DJ table which would return a list of the DJ’s mentors
  + CREATE INDEX

IDX\_MENTOR\_ID ON dj

(MANID);

* IDX\_SHOW\_TIME that would return ShowDay and ShowTime for the different shows in the system
  + CREATE INDEX

IDX\_SHOW\_TIME ON showdesc

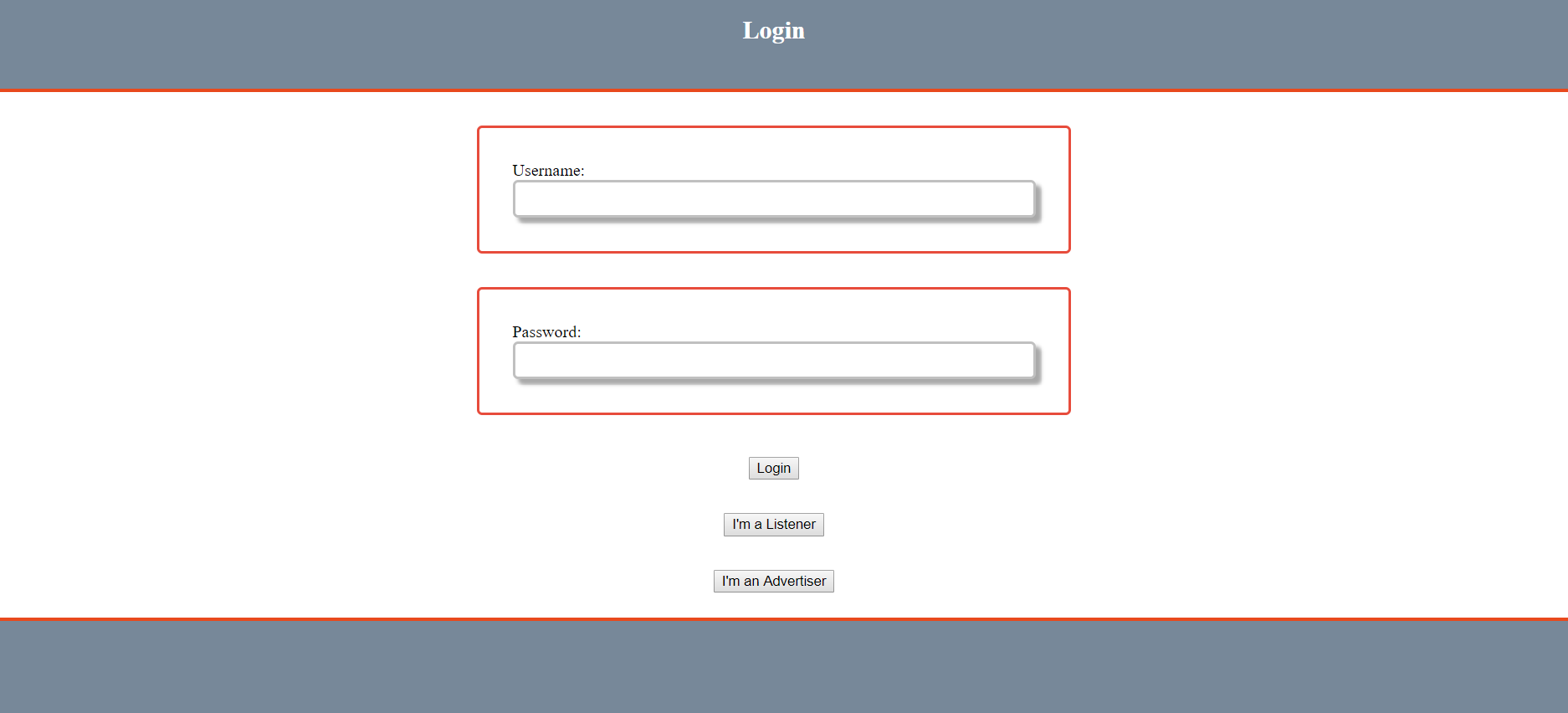
(ShowDay, Showtime);

* IDX\_SONGNAMES which would return SongNames, rather than their additional, unnecessary details.
  + CREATE INDEX

IDX\_SONGNAMES ON songRequest

(SongName);

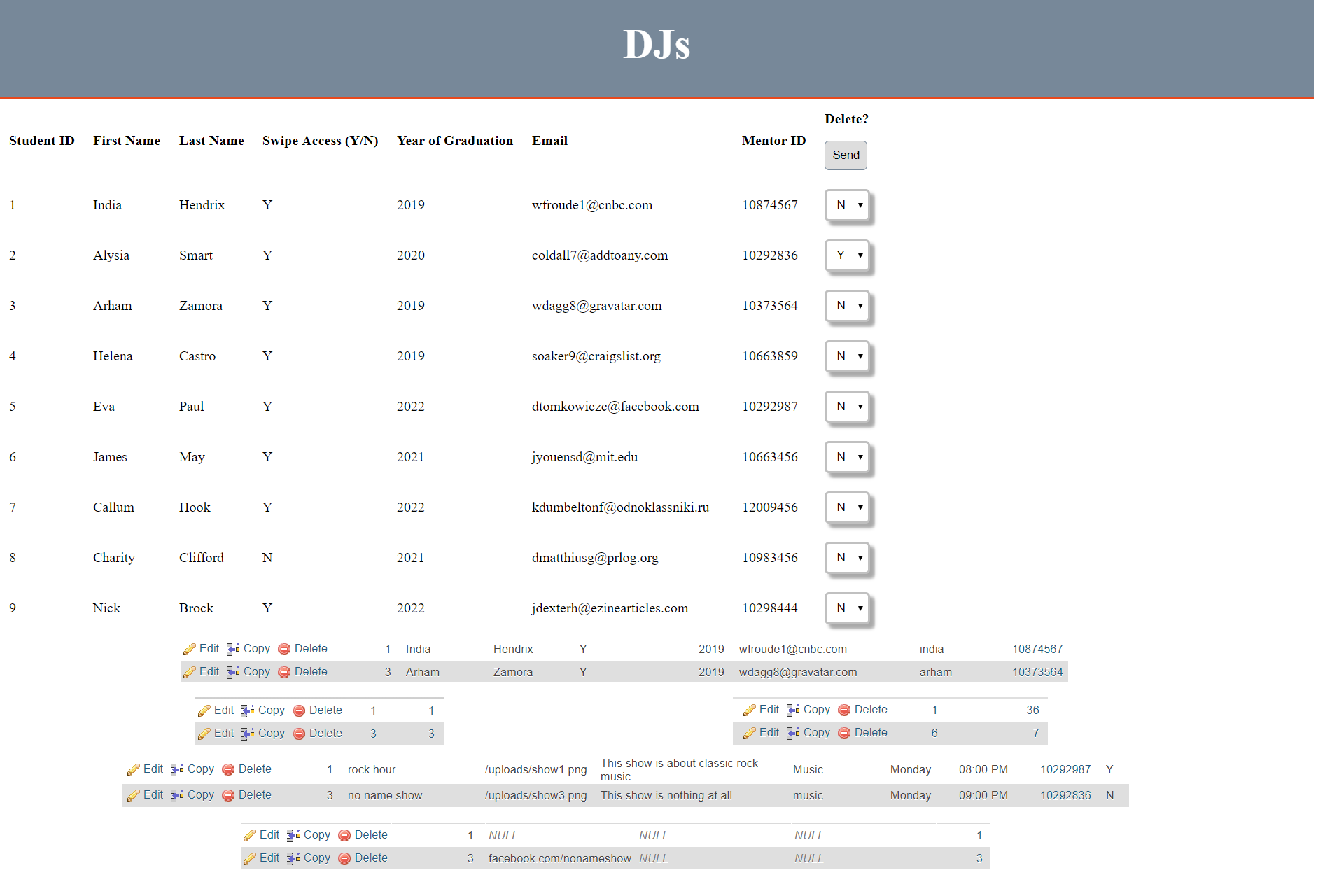
Images from the Database / PHP Code:



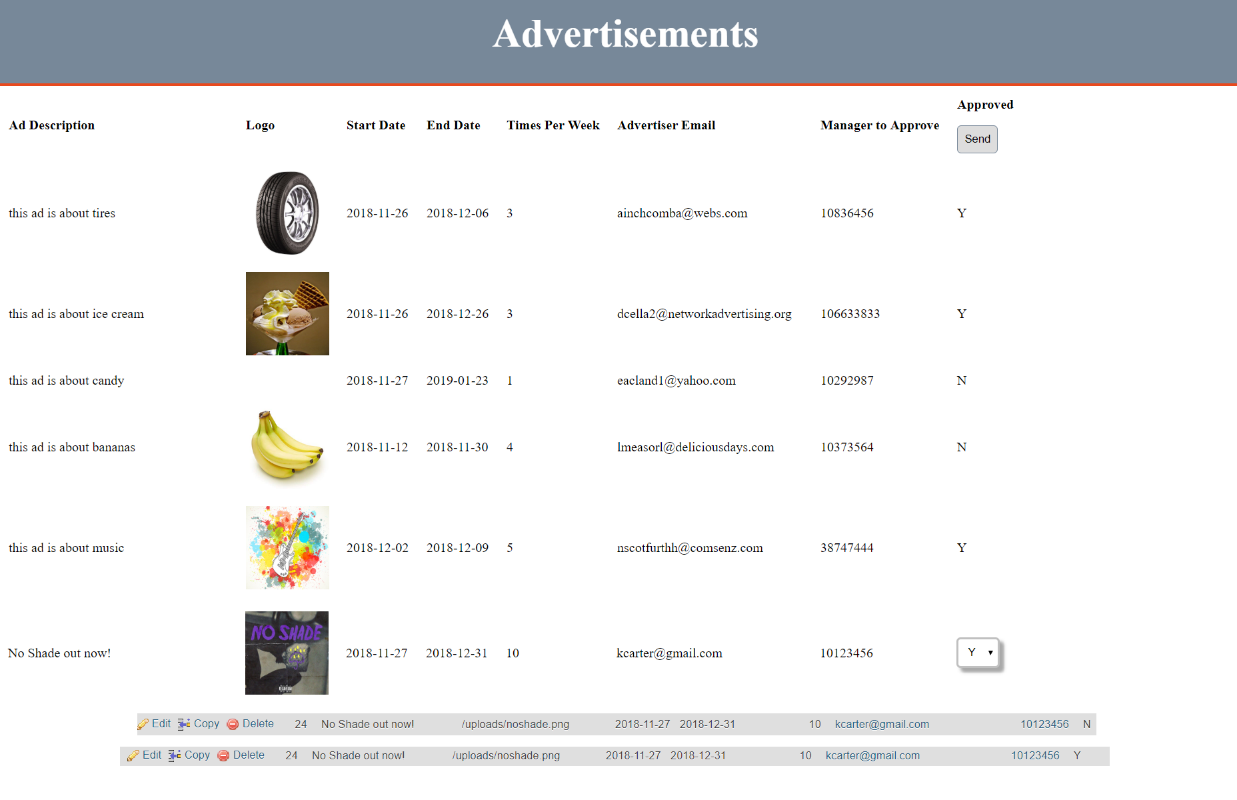
MANAGER:



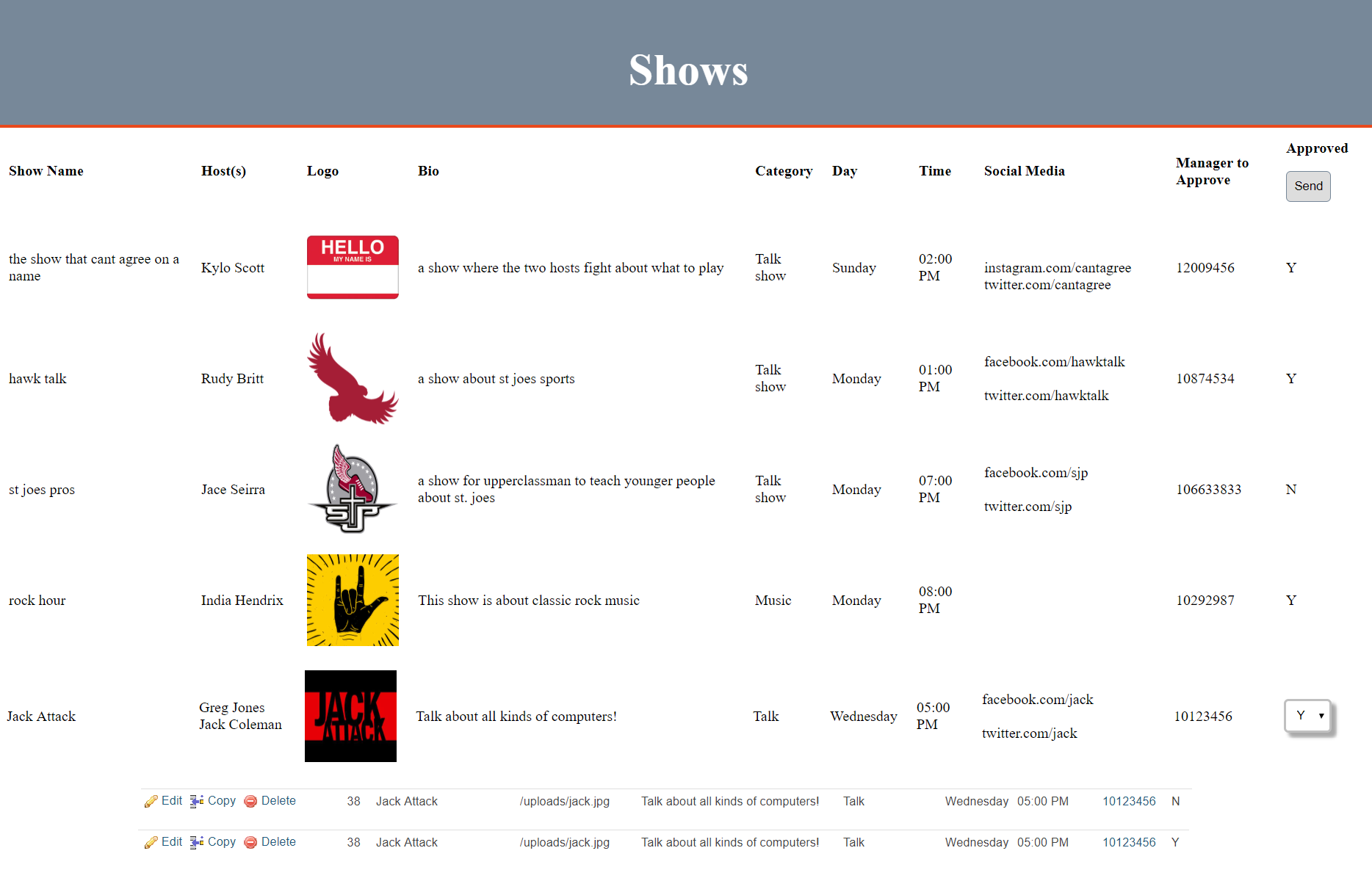




Here, the Manager can see DJs and can delete them from the system (and anywhere their ID is located)



Here, the Manager can see ads that are either approved or unapproved, and can approve ads that are designated to their ID



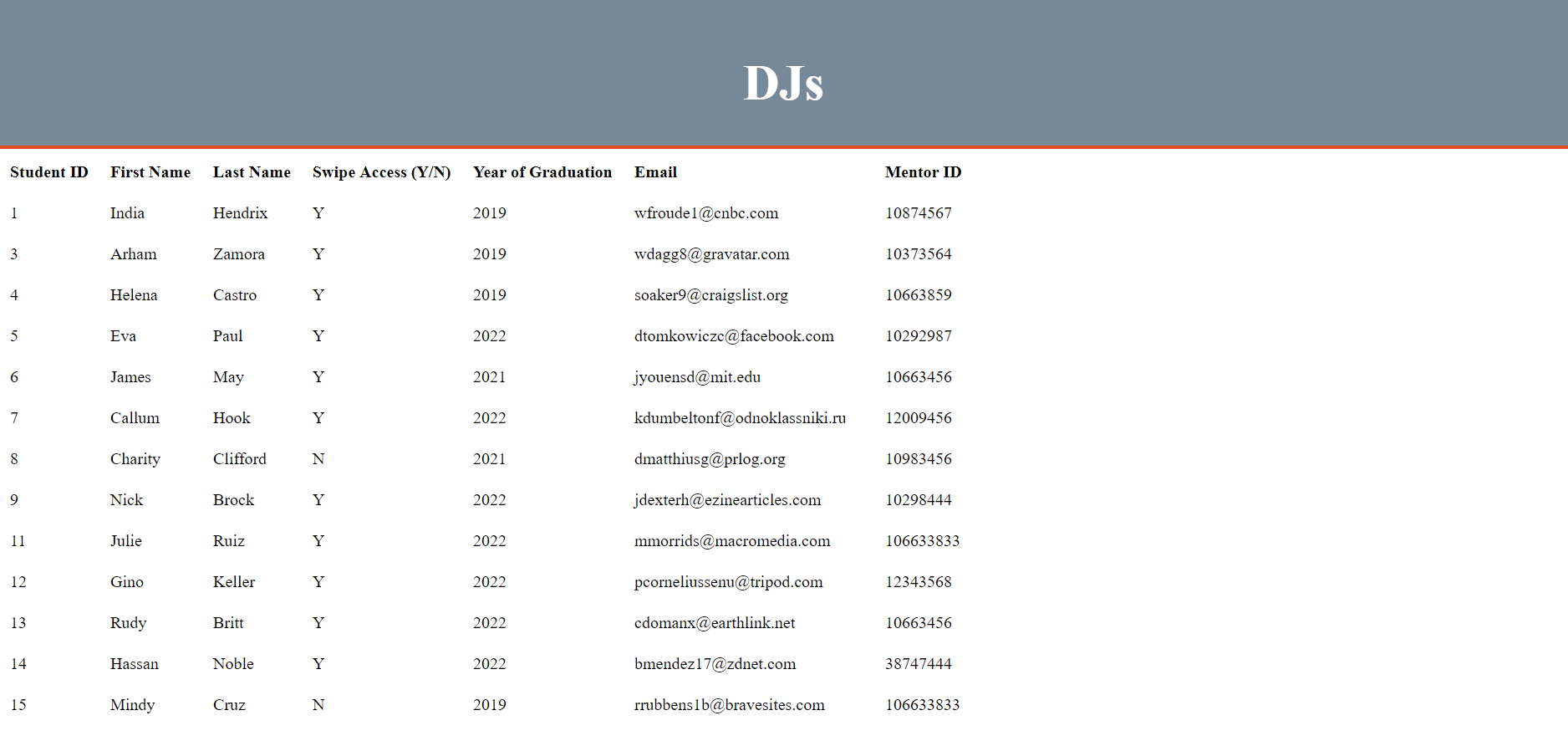
Here, the Manager can see shows that are either approved or unapproved, and can approve shows that are designated to their ID

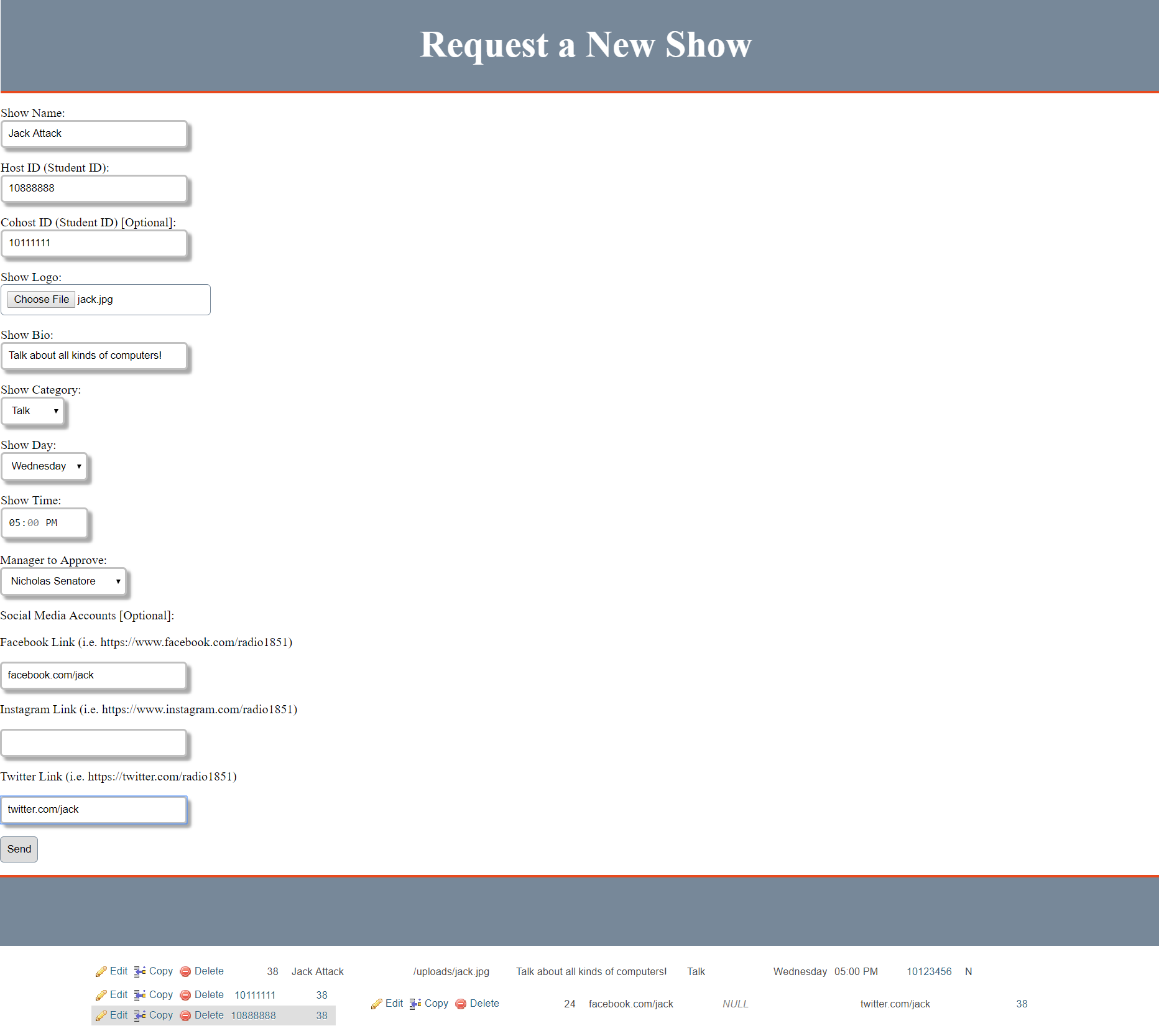


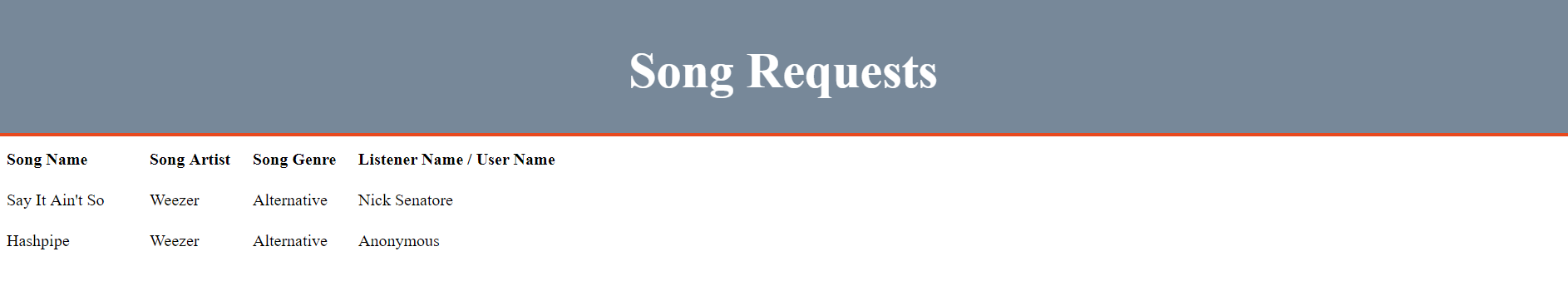
Here, the Manager can see songs requested for ANY show

DJ:









Here, the DJ can only see songs requested for HIS show

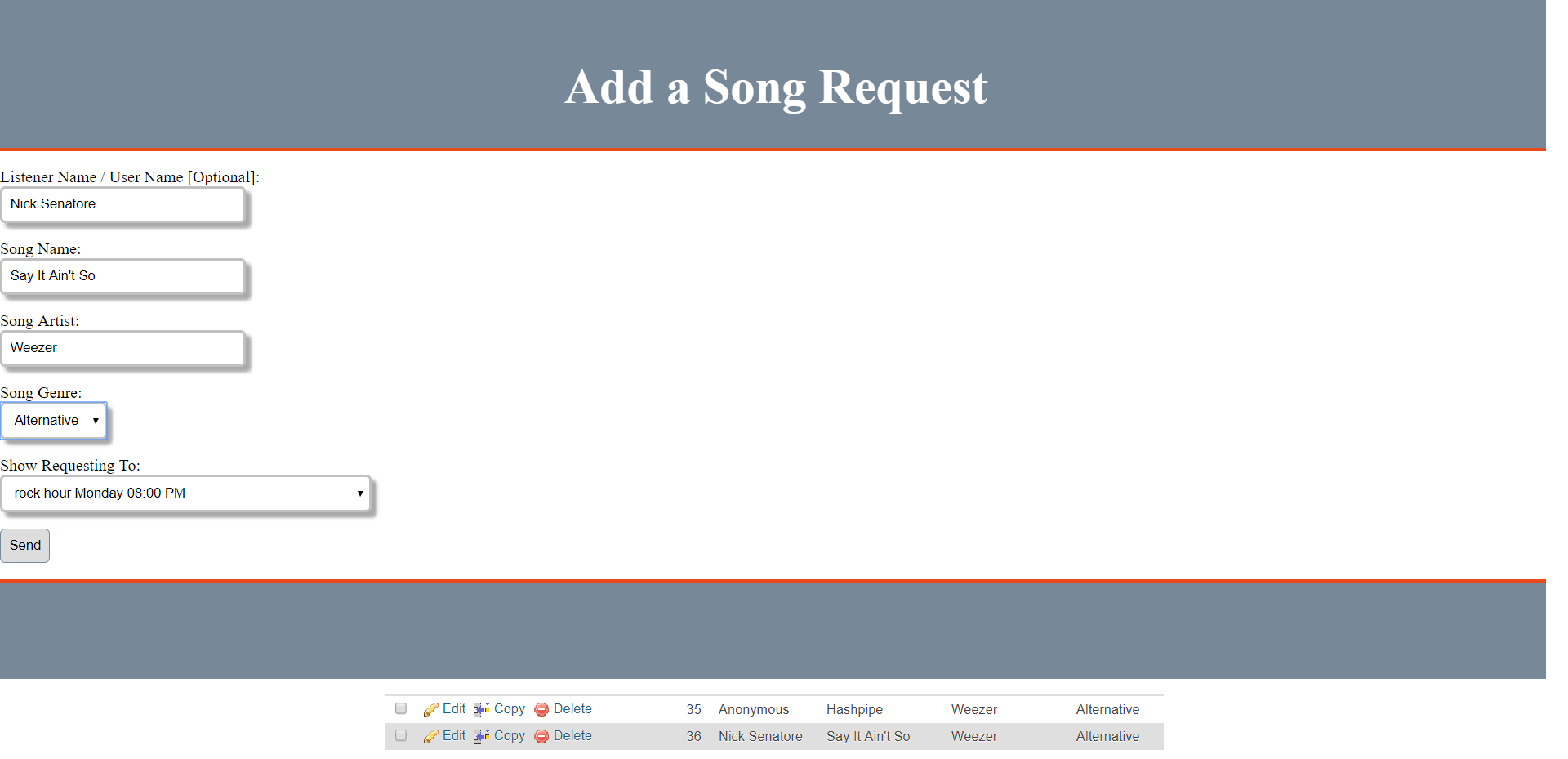
Advertiser:



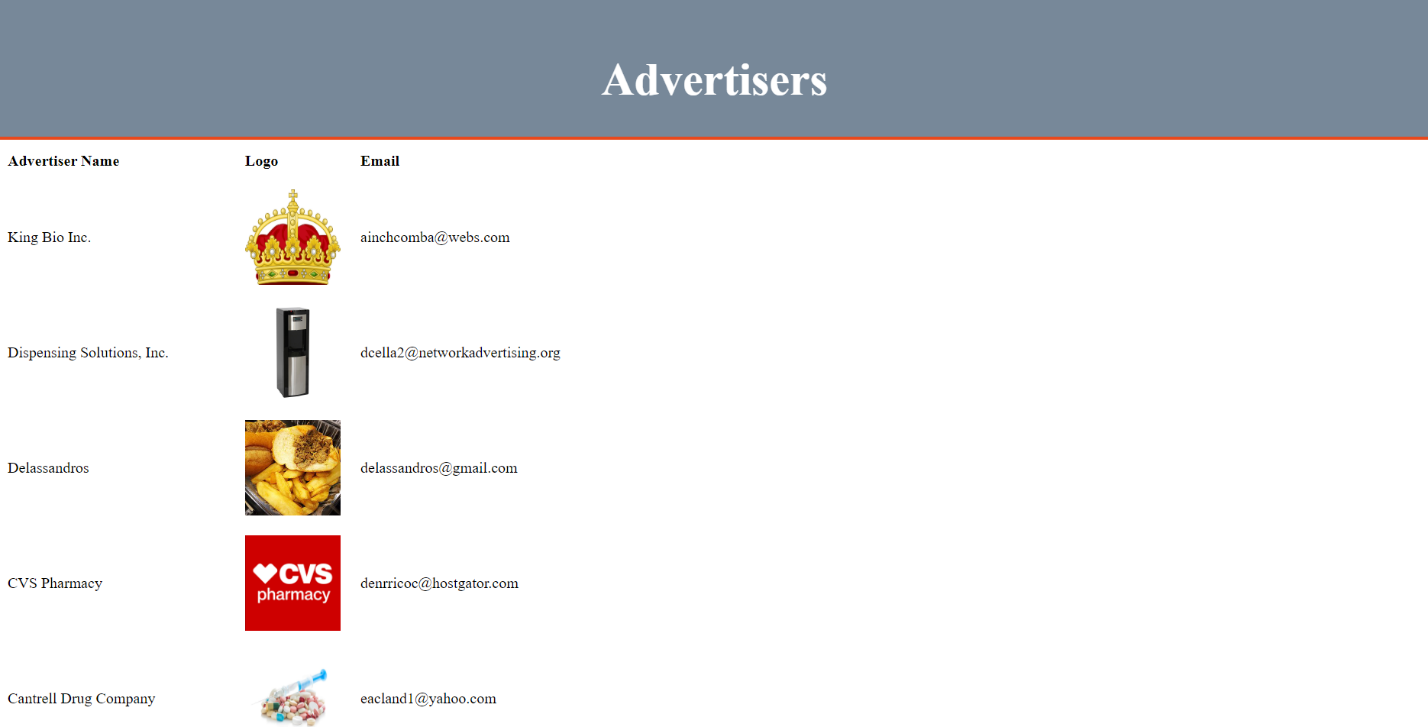


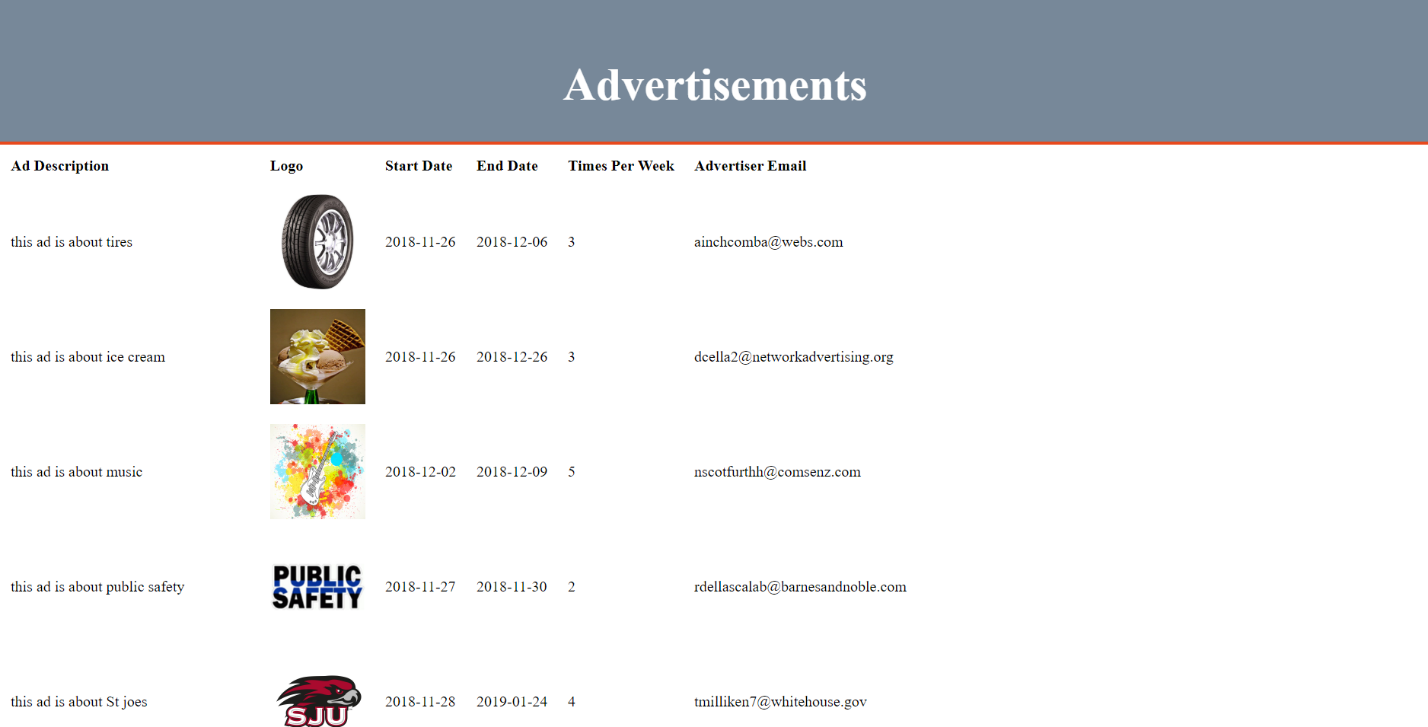
You need to be registered as an advertiser in order to request an ad

Listener:

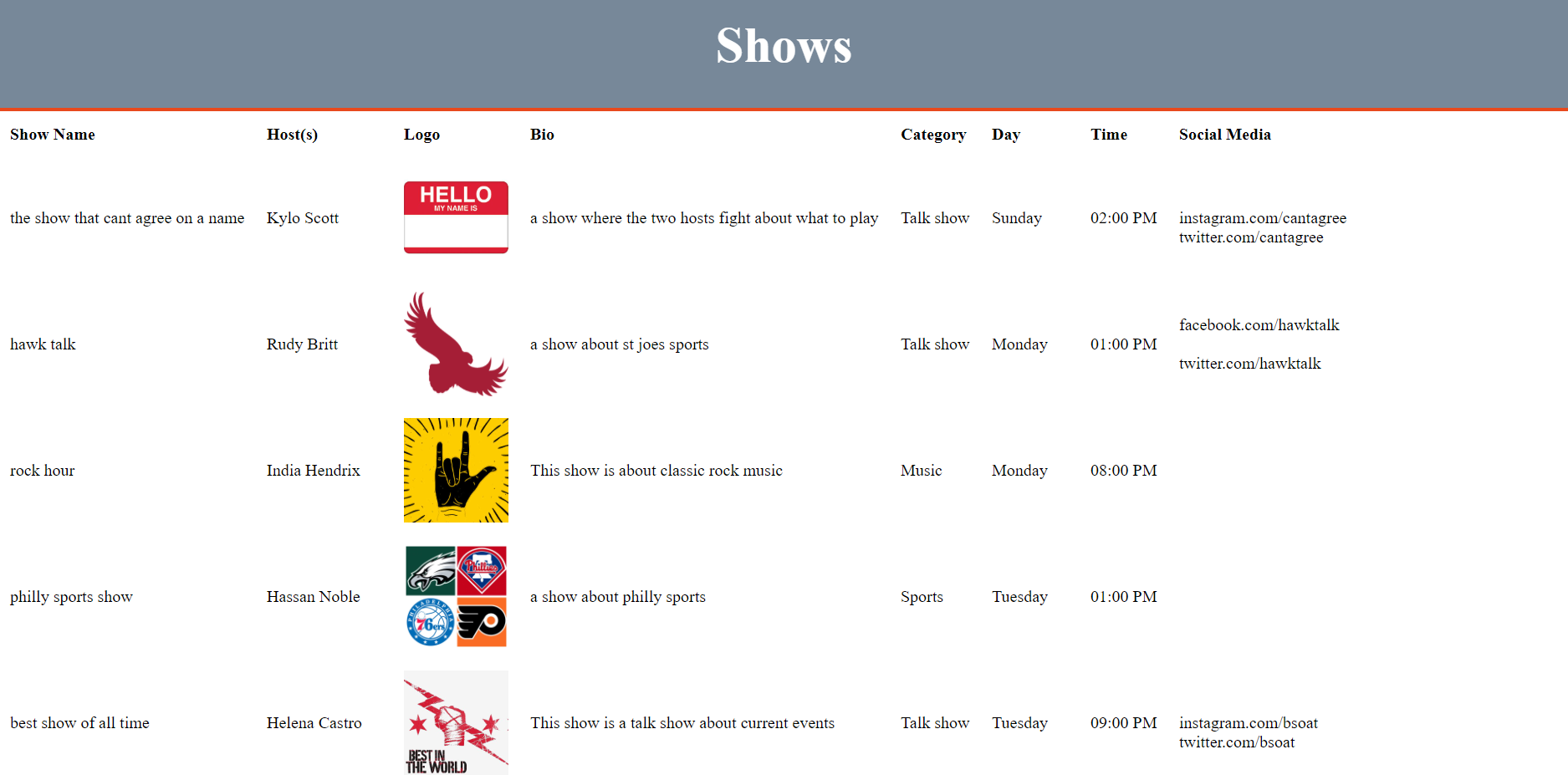


If someone does not want to disclose their name, a DEFAULT value of “Anonymous” will be sent for Name





Here, the Listener (and DJs / Advertisers) can ONLY see ads that are APPROVED



Here, the Listener (and DJs / Advertisers) can ONLY see shows that are APPROVED

Checking / Constraints on Update, Delete:

Swipe Access was limited to Yes or No, while Year Graduated was allowed only from the current year up to about five years from the current year. Start Dates for advertisements can not be before the current date. DJs can only have one hour slots, with only the hour being adjustable. DJs can also only host shows between a given time period (9:00AM – 11:00PM). Delete would only occur on the given DJ that was chosen to be deleted, and all of his or her respective child references would be removed, as well.

Useful SQL Code:

* Determine how many advertisers have ads
  + SELECT DISTINCT AV.AdvertiserName FROM advertiser AV, ad A WHERE AV.AdvertiserEmail = A.AdvertiserEmail;
* Get the DJ Full Name and ID for DJs that host or cohost a show
  + SELECT D.DJID, CONCAT(D.DJFirstName, ' ', D.DJLastName) AS DJName FROM dj D, hosts H WHERE D.DJID = H.DJID;
* Get the Show Bio for all shows that are not yet approved (For fast approval / overview)
  + SELECT ShowName, ShowBio FROM showdesc WHERE ShowApproved = 'N';
* Determine the social media accounts for all shows, side by side
  + SELECT S.ShowName, SM.Facebook, SM.Instagram, SM.Twitter FROM showdesc S, socialmedia SM WHERE S.ShowID = SM.ShowID;
* Get the Song Name for all Song Requests that are not submitted anonymously
  + SELECT SongName FROM songrequest WHERE ListenerName != "Anonymous";
* Determine how many song requests each show has
  + SELECT ShowID, COUNT(\*) AS numRequests FROM views GROUP BY ShowID;

Extra Credit:

* Images were used in the database. To try them out yourself, download the images from the folder in the Git Repository.
* Special users were defined for Managers, DJs, Listeners, and Advertisers; each with specific abilities. SESSION was used to check if the current user was allowed to access a given .PHP file.
* Logons for different users were created (Managers, DJs, Advertisers), each with specific abilities. Managers could add managers, DJs, approve shows and ads. DJs could view song requests and request shows. Advertisers could be registered and request advertisements.
* We used Git in order to share files with one another throughout the process. Different versions were created for various files, and test files were sent to one another.   
  GitHub Repository - <https://github.com/nicksters318/radio1851Database>