Web API Documentation for FindAGig Final Project

https://localhost:8080/

Http Verb Method	Endpoint	Description
POST	/users	CREATE: Store a new user Needs firstName, lastName, userType, instruments, & address *(creates a user & address (if needed) & instrument (if needed) & musician_instrument record)*
GET	/users	READ: Get all users — MUSICIANS & PLANNERS
GET	/users/{id}	READ: Get one user by {id}
PUT	/users/{id}	UPDATE: Update a user by {id} — Needs all fields
DELETE	/users/{id}	DELETE: Delete a user by {id} **FUTURE:** DELETE: Delete a user by {id}. **User must be SystemAdmin!**
GET	/users/{id}/gigs	READ: Get all positions that match user by {id} — **FUTURE:** **Print out Gig information for each requested position.**
POST	/instruments	CREATE: Stores an instrument — Needs a name. ***(creates an instrument record)***
GET	/instruments	READ: Get all instruments
PUT	/instruments/{id}	UPDATE: Update an instrument by {id} — Needs a name
DELETE	/instruments/{id}	This is NOT ALLOWED! **FUTURE:** DELETE: Delete an instrument by {id}. **User must be SystemAdmin!**
POST	/gigs	CREATE: Stores a gig — Needs date, phone, event, genre, plannerId description, salary *(creates a gig record only — instruments must be added later!)* **FUTURE:** **Add Instruments as part of this operation if specified.**
GET	/gigs	READ: Gets all gigs

Http Verb Method	Endpoint	Description
GET	/gigs/users/{id}	READ: Get all gigs that match a particular user — the user has REQUESTED or is CONFIRMED for a particular gig! **FUTURE:** **Add Gig Event Info as well!**
GET	/gigs/{id}	READ: Get all instruments required for a gig {id}. *This prints out Gig & Instrument information.*
POST	/gigs/{id}	CREATE: Adds Instruments to a gig by {id} — *(creates gig_status & gig_status_instrument records)*
PUT	/gigs/{id}	UPDATE: Update a gig by {id}. **Instruments** (GigStatus records) status is changed as well if new Status is NOT OPEN. If new Status is OPEN, the instruments also need to remain the same! **FUTURE:** **Add more robust handling of the instrument records.**
DELETE	/gigs/{id}	This is NOT ALLOWED! **Gig Planner must be SystemAdmin!** **FUTURE:** DELETE: Delete a gig by {id}. **User must be SystemAdmin!**
PUT	/gigs/{id}/status/{statusType}	UPDATE: Change ONLY the status for a GIG and attached Gig Status Records. Change Status from one to another! **PLANNED** to **OPEN** to… to **CANCELLED** or **CLOSED**.
GET	/gigs/{id}/users	READ: Get all users that match a particular gig by {id} — Print Musician Information — **FUTURE: PRINT Gig Information**

Http Verb Method	Endpoint	Description

PUT	/gigs/{id}/users/{id}/request	UPDATE: Update a gig {id} to “REQUESTED” a user by {id}
		
		{
		"instruments" :[{
		"name" : "Trumpet"
		}],
		"status" : "REQUESTED"
		}

PUT	/gigs/{id}/users/{musicianId}/confirm/{plannerId}	UPDATE: Update a gig {id} to “CONFIRM” a musician {musicianId} — DONE by the gig planner {plannerId}
		
		{
		"instruments" :[{
		"name" : "Piano"
		}],
		"status" : "CONFIRMED"
		}

GET	/gigs/open	READ: all gigs with OPEN positions
GET	/gigs/open/instrument/{instName}	READ: all gigs OPEN positions by particular Instrument
GET	/gigs/open/state/{stateName}	READ: all gigs OPEN positions in a particular state
GET	/gigs/open/genre/{genreName}	READ: all gigs OPEN positions by particular genre
GET	/gigs/instrument/{instName}	READ: all gigs using a particular instrument
GET	/gigs/state/{stateName}	READ: all gigs in a particular state
GET	/gigs/genre/{genreType}	READ: all gigs by particular genre