**Steps To Create the REST API**

1. We created our empty **Schema** on GearHost, then specified it on our .YAML file.
2. Created all needed **models**: Address, Client, Instructor, Location, Resource, Student, Tool, User, and UserType. All are annotated by @Entity and use **foreign keys**, except Address.
3. We needed Relational Tables with working **multiplicity** in MySQL.
   1. Using Hibernate, we declared our relational tables in our models by using our **annotations** such as @OneToOne or @OneToMany **multiplicity**.
   2. Each table generated with Hibernate lacks a primary key and is made up of **foreign keys**. We used these **foreign keys** to pair certain table rows.
4. Added Hibernate as a POM.xml dependency, then used its **annotations**.
5. Properly filled out all model files.
   1. All private values have @Column **annotations**, Getter and Setter methods, and two Constructors: empty and filled.
   2. The id column required @Id and @GeneratedValue **annotations**.
6. Made our @Repository **marker interfaces** to bridge access to our database.
7. Created our @Service layer classes to hold logic of our CRUD operations.
8. Chose not to use DTO or Mappers as that isn’t suited for our structural style.
9. Built our @RestController classes to carry out HTTP requests and return valid responses to clients.
10. Added CORS configuration class (Cross-Origin Resource Sharing) to facilitate communication between our Spring Boot project and our Angular directory.
11. Performed unit testing using dummy data, posting data such as Client data and receiving both its data and any **foreign key** data.
    1. For instance, Client has **foreign key** data: an Address object. When performing a GET request, it displays both Client and Address. Which Address is displayed depends on the paired **foreign keys** table.