

3rd YEAR TEST-10 TOTAL TIME → 1.5hrs

Below is the representation of the task that you need to perform.

- let's say your University roll number is 291029063
- your set = (University roll number) % 15 = 291029063 % 15 = 8
- so your set number is 8
- so you have to follow the below question set-8

Question Set - 0

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/person/:id', '/person/:id/edit', '/person' and '/person/:id/delete'.
(use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only person name cannot be changed, rest of everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11Do not share the node_modules and package-lock.json.

Person (model name)
name <input checked="" type="checkbox"/>
age <input checked="" type="checkbox"/>
email <input checked="" type="checkbox"/>
D.O.B <input checked="" type="checkbox"/>

personSchema (schema name)

name: → type , mandatory , trim
age: → type , mandatory , trim
email: → type , mandatory , trim
D.O.B: → type , mandatory

Question Set - 1






1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/product/:id', '/product/:id/edit', '/product' and '/product/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only productName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Product (model name)
name <input checked="" type="checkbox"/>
price <input checked="" type="checkbox"/>
category <input checked="" type="checkbox"/>
quantityInStock <input checked="" type="checkbox"/>

productSchema (schema name)
name: → type , mandatory , trim price: → type , mandatory , trim category: → type , mandatory , trim quantityInStock: → type , mandatory

Question Set - 2






1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) ‘/blog/:id’ , ‘/blog/:id/edit’ , ‘/blogs and ‘/blog/:id/delete’ (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only blogTitle cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Blog (model name)
blogTitle  description  createdAt  author  ImageUrl 

blogSchema (schema name)
blogTitle : → type , mandatory , trim description : → type , mandatory , trim createdAt : → type , mandatory , trim author : → type , mandatory imageUrl : → type , mandatory , trim

Question Set - 3

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/song/:id', '/song/:id/edit', '/songs' and '/song/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only songTitle cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Song (model name)
songTitle  singer  genre  releaseDate  songImageUrl 

songSchema (schema name)
songTitle : → type , mandatory , trim singer: → type , mandatory , trim genre: → type , mandatory , trim releaseDate: → type , mandatory , Date songImageUrl: → type , mandatory

Question Set - 4






1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/contact/:id', '/contact/:id/edit', '/contacts' and '/contact/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only contactName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Contact (model name)
contactName <input checked="" type="checkbox"/> phoneNo <input checked="" type="checkbox"/> email <input checked="" type="checkbox"/> imageUrl <input checked="" type="checkbox"/>

contactSchema (schema name)
contactName : → type , mandatory , trim phoneNo: → type , mandatory , trim email: → type , mandatory , trim imageUrl: → type , mandatory

Question Set - 5

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/realEstate/:id', '/realEstate/:id/edit', '/realEstates and '/realEstate/:id/delete'(use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only propertyName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

RealEstate (model name)
propertyName  owner  totalDimension  Location  imageUrl 

realestateSchema (schema name)
propertyName : → type , mandatory , trim owner: → type , mandatory , trim totalDimension : → type , mandatory , trim Location : → type , mandatory imageUrl : → type , mandatory

Question Set - 6

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/event/:id', '/event/:id/edit', '/event' and '/event/:id/delete'(use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only eventName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Event (model name)
name <input checked="" type="checkbox"/> location <input checked="" type="checkbox"/> date <input checked="" type="checkbox"/> attendees <input checked="" type="checkbox"/>

eventSchema (schema name)
name: → type , mandatory , trim location: → type , mandatory , trim date: → type , mandatory , trim attendees: → type , mandatory

Question Set - 7

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed)'/car/:id', '/car/:id/edit', '/car' and '/car/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only car company cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Car (model name)

company ✓
model ✓
year ✓
mileage ✓

carSchema (schema name)

company: → type , mandatory , trim
model: → type , mandatory , trim
year: → type , mandatory , trim
mileage: → type , mandatory

Question Set - 8

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/music-album/:id', '/music-album/:id/edit', '/music-album' and '/music-album/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only musicTitle cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Music (model name)

musicTitle ✓
artist ✓
genre ✓
releaseYear ✓

musicSchema (schema name)

musicTitle: → type , mandatory , trim
artist: → type , mandatory , trim
genre: → type , mandatory , trim
releaseYear: → type , mandatory

Question Set - 9

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/mobile-phone/:id', '/mobile-phone/:id/edit', '/mobile-phone' and '/mobile-phone/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only brand cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Mobile (model name)

brand ✓
model ✓
operatingSystem ✓
storageCapacity ✓
releaseDate ✓

mobileSchema (schema name)

brand: → type , mandatory , trim
model: → type , mandatory , trim
operatingSystem: → type , mandatory , trim
storageCapacity: → type , mandatory
releaseDate: → type , mandatory , trim

Question Set - 10

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/university/:id', '/university/:id/edit', '/universities' and '/university/:id/delete'(use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only universityName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

University (model name)

universityName ✓
affiliateBy ✓
location ✓
numberOfCourses ✓
ImageUrl ✓

mobileSchema (schema name)

universityName : → type , mandatory , trim
affiliateBy: → type , mandatory , trim
location: → type , mandatory , trim
numberOfCourses: → type , mandatory
ImageUrl: → type , mandatory , trim

Question Set - 11

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/restaurant/:id', '/restaurant/:id/edit','/restaurants' and '/restaurant/:id/delete'(use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only restaurantName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Restaurant (model name)

restaurantName ✓

Cuisine ✓

Chef ✓

phoneNo ✓

ImageUrl ✓

restaurantSchema (schema name)

restaurantName : → type , mandatory , trim

Cuisine : → type , mandatory , trim

Chef : → type , mandatory , trim

phoneNo : → type , mandatory

ImageUrl : → type , mandatory , trim

Question Set - 12

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/airline/:id', '/airline/:id/edit', '/airlines' and '/airline/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only companyName be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Airline (model name)

companyName ✓
owner ✓
numberOfPlane ✓
planeCapacity ✓
Model ✓

airlineSchema (schema name)

companyName : → type , mandatory , trim
owner: → type , mandatory , trim
numberOfPlane : → type , mandatory , trim
planeCapacity : → type , mandatory
Model : → type , mandatory , trim

Question Set - 13

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/hotel/:id', '/hotel/:id/edit', '/hotels' and '/hotel/:id/delete'(use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only hotelName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Hotel (model name)

hotelName ✓
rating ✓
numberOfRoom ✓
Location ✓
roomImageUrl ✓

hotelSchema (schema name)

hotelName : → type , mandatory , trim
rating: → type , mandatory , trim
numberOfRoom : → type , mandatory , trim
Location : → type , mandatory
roomImageUrl : → type , mandatory , trim

Question Set - 14

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform READ , EDIT , SHOW, DELETE functionality. (use forms wisely if needed) '/zoo/:id', '/zoo/:id/edit', '/zoo' and '/zoo/:id/delete' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only zooName cannot be changed, rest everything can change.
5. Follow proper folder structures
6. Public → style according to the template shown (just for reference)
7. Views → create proper template files
8. Models → make different schema
9. Routes → make different routing files
- 10.No need to have any dummy data.
- 11.Do not share the node_modules and package-lock.json.

Zoo (model name)

zooName ✓

numberOfAnimal ✓

location ✓

openingDuration ✓

zooImageUrl ✓

zoomSchema (schema name)

zooName: → type , mandatory , trim

numberOfAnimal: → type , mandatory , trim

location: → type , mandatory , trim

openingDuration: → type , mandatory

zooImageUrl: → type , mandatory , trim