

### 3rd YEAR TEST-10

TOTAL TIME → 1.5hrs

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

→ let's say your University roll number is 2156763

→ your set = (University roll number)%15= 2156763%15=3

→ so your set number is 3

→ so you have to follow the below question set-3

#### QUESTION SET - 0

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform **READ , EDIT , SHOW** functionality. (use forms wisely if needed)  
**‘/book/:id’ , ‘/book/:id/edit’ , ‘/book’** (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only **bookTitle** 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.

Book (model name)
title <input checked="" type="checkbox"/> author <input checked="" type="checkbox"/>
genre <input checked="" type="checkbox"/>
publicationYear <input checked="" type="checkbox"/>
bookTitle <input checked="" type="checkbox"/> price
<input checked="" type="checkbox"/> timestamps <input checked="" type="checkbox"/>

bookSchema (schema name)
--------------------------

title:

→ type , mandatory , trim author:

→ type , mandatory , trim genre:

→ type , mandatory , trim

publicationYear: → type ,

mandatory bookTitle:

→ type , mandatory , trim price:

→ type , mandatory

timestamps: →

created time

## QUESTION SET - 1

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform **READ , EDIT , DELETE** functionality. (use forms wisely if needed) **'/recipes' , '/recipes/:id/edit' , '/recipes/:id/delete'** (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only **recipeName** 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.

Recipe (model name)

name ✓ chef

✓ cuisine ✓

ingredients ✓

recipeName ✓

price ✓

timestamps ✓

recipeSchema (schema name)

## QUESTION SET - 2

1. Create a schema below).
2. Perform

name:

- type , mandatory , trim chef:
- type , mandatory , trim cuisine:
- type , mandatory , trim ingredients:
- type , mandatory , trim recipeName:
- type , mandatory , trim price:
- type , mandatory timestamps:
- created time

model according to the following diagrams (box shown

**READ , EDIT , DELETE** functionality. (use forms wisely if needed)  
'/movies' , '/movies/new' , '/movies/:id/delete' (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **movieTitle** 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.

Movie (model name)

title ✓ director  
✓ genre ✓  
releaseYear ✓  
movieTitle ✓  
price ✓  
timestamps ✓

movieSchema (schema name)

title:

- type , mandatory , trim director:
- type , mandatory , trim genre:
- type , mandatory , trim releaseYear:
- type , mandatory movieTitle:
- type , mandatory , trim price:
- type , mandatory

timestamps: →

created time

### QUESTION SET - 3

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform **READ , EDIT , DELETE** functionality. (use forms wisely if needed)  
    '**/shoes**' , '**/shoes/:id**' , '**/shoes/:id/delete**' (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only **shoeModel** cannot be changed, rest everything can change.
5. Follow proper folder structures
6. **Public** → style according to the template shown (just for reference)

#### QUESTION SET - 4

1. Create a schema below).
2. Perform
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.

Shoe (model name)

brand ☒ color

☒ size ☒

shoeModel ☒

price ☒

timestamps ☒

shoeSchema (schema name)

brand:

→ type , mandatory , trim color:

→ type , mandatory , trim size:

→ type , mandatory shoeModel:

→ type , mandatory , trim price:

→ type , mandatory

timestamps: →

created time

**CREATE , SHOW & DELETE** functionality. (use forms wisely if needed)

**‘/laptops’ , ‘/laptops/new’ , ‘/laptops/:id’ , ‘/laptops/:id/delete’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **laptopModel** 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.

Laptop (model name)
brand <input checked="" type="checkbox"/> processor
<input checked="" type="checkbox"/> ramSize <input checked="" type="checkbox"/>
laptopModel <input checked="" type="checkbox"/>
price <input checked="" type="checkbox"/>
timestamps <input checked="" type="checkbox"/>

laptopSchema (schema name)
title:
→ type , mandatory , trim director:
→ type , mandatory , trim genre:
→ type , mandatory , trim releaseYear:
→ type , mandatory movieTitle:
→ type , mandatory , trim price:
→ type , mandatory
timestamps: →
created time

## QUESTION SET - 6

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform

## QUESTION SET - 5

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform **CREATE , EDIT & DELETE** functionality. (use forms wisely if needed)  
**‘/shirts’ , ‘/shirts/new’ , ‘/shirts/:id/edit’ , ‘/shirts/:id/delete’** (use proper routing convention)
3. Form should have inputs according to the table below.
4. # Only **shirtColor** 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.

Shirt (model name)
fabric <input checked="" type="checkbox"/> size
<input checked="" type="checkbox"/> pattern <input checked="" type="checkbox"/>
shirtColor <input checked="" type="checkbox"/>
price <input checked="" type="checkbox"/>
timestamps <input checked="" type="checkbox"/>

shirtSchema (schema name)
---------------------------

fabric:

- type , mandatory , trim size:
- type , mandatory , trim pattern:
- type , mandatory , trim shirtColor:
- type , mandatory , trim price:
- type , mandatory

timestamps: →

created time

**CREATE , EDIT & DELETE** functionality. (use forms wisely if needed)

**‘/cameras/new’ , ‘/cameras/:id/edit’ , ‘/cameras/:id/delete’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **cameraModel** 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.

Camera (model name)

brand ☒ resolution  
☒ megapixels ☒  
cameraModel ☒  
price ☒ timestamps  
☒

cameraSchema (schema name)



## QUESTION SET - 8

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform

brand:

- type , mandatory , trim resolution:
- type , mandatory , trim megapixels:
- type , mandatory cameraModel:
- type , mandatory , trim price:
- type , mandatory

timestamps: →

created time

## QUESTION SET - 9

1. Create a schema below).
2. Perform

model according to the following diagrams (box shown

**CREATE , SHOW & DELETE** functionality. (use forms wisely if needed)

**‘/guitars’ , ‘/guitars/:id’ , ‘/guitars/:id/delete’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **guitarBrand** 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.

Guitar (model name)
type <input checked="" type="checkbox"/> color
<input checked="" type="checkbox"/> material <input checked="" type="checkbox"/>
guitarBrand <input checked="" type="checkbox"/>
price <input checked="" type="checkbox"/>
timestamps <input checked="" type="checkbox"/>

guitarSchema (schema name)
type:
→ type , mandatory , trim color:
→ type , mandatory , trim material:
→ type , mandatory , trim guitarBrand:
→ type , mandatory , trim price:
→ type , mandatory
timestamps: →
created time

**READ , SHOW & EDIT** functionality. (use forms wisely if needed)

**‘/watches’ , ‘/watches/:id’ , ‘/watches/:id/edit’** (use proper routing convention)

## QUESTION SET - 10

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform
3. Form should have inputs according to the table below.
4. # Only **watchBrand** 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.

Watch (model name)

brand  type   
strapMaterial   
watchBrand   
price   
timestamps 

watchSchema (schema name)

brand:  
→ type , mandatory , trim type:  
→ type , mandatory , trim strapMaterial:  
→ type , mandatory , trim watchBrand:  
→ type , mandatory , trim price:  
→ type , mandatory  
timestamps: →  
created time

## QUESTION SET - 11

1. Create a schema below).
2. Perform

model according to the following diagrams (box shown

**CREATE, EDIT, SHOW & DELETE** functionality. (use forms wisely if needed)

**‘/guns, ‘/guns/new’, ‘/guns/:id/edit’, ‘/guns/:id/delete’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **gunName** cannot be changed, rest everything can change.
5. Add a new feature - **bulletType**
6. Follow proper folder structures
7. **Public** → style according to the template shown (just for reference)
8. **Views** → create proper template files
9. **Models** → make different schema
10. **Routes** → make different routing files
11. No need to have any dummy data.
12. Do not share the node\_modules and package-lock.json.

Instrument (model name)
type <input checked="" type="checkbox"/> brand <input checked="" type="checkbox"/> gunName <input checked="" type="checkbox"/> price <input checked="" type="checkbox"/> bulletType <input checked="" type="checkbox"/> timestamps <input checked="" type="checkbox"/>

instrumentSchema (schema name)
type: → type, mandatory, trim brand: → type, mandatory, trim gunName: → type, mandatory, trim price: → type, mandatory bulletType: → type, mandatory timestamps: → created time

## QUESTION SET - 12

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform

**CREATE, SHOW & DELETE** functionality. (use forms wisely if needed)

**‘/electronics’, ‘/electronics/new’, ‘/electronics/:id/delete’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **electronicName** 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.

Electronics (model name)
category <input checked="" type="checkbox"/> brand
<input checked="" type="checkbox"/> electronicName
<input checked="" type="checkbox"/> price <input checked="" type="checkbox"/>
warrantyPeriod <input checked="" type="checkbox"/>
timestamps <input checked="" type="checkbox"/>

electronicsSchema (schema name)
---------------------------------

## QUESTION SET - 13

1. Create a schema below).
2. Perform

category:

- type, mandatory, trim brand:
- type, mandatory, trim electronicName:
- type, mandatory, trim price:
- type, mandatory

warrantyPeriod: →

type, mandatory

timestamps: →

created time

## QUESTION SET - 14

1. Create a schema-model according to the following diagrams (box shown below).
2. Perform

**CREATE, EDIT, SHOW** functionality. (use forms wisely if needed)  
**‘/instruments’, ‘/instruments/new’, ‘/instruments/:id ’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **instrumentName** 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.

Instrument (model name)
type <input checked="" type="checkbox"/> brand <input checked="" type="checkbox"/> instrumentName <input checked="" type="checkbox"/> price <input checked="" type="checkbox"/> soundQuality <input checked="" type="checkbox"/> timestamps <input checked="" type="checkbox"/>

instrumentSchema (schema name)
type: → type, mandatory, trim brand: → type, mandatory, trim instrumentName: → type, mandatory, trim price: → type, mandatory soundQuality: → type, mandatory timestamps: → created time

## QUESTION SET - 15

1. Create a schema-model according to the following diagrams (box shown below).
- 2.

Perform **CREATE, EDIT & DELETE** functionality. (use forms wisely if needed)

**‘/vehicles/new’, ‘/vehicles/:id/edit’, ‘/vehicles/:id/delete’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **vehicleModel** 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.

Vehicle (model name)
type <input checked="" type="checkbox"/> brand <input checked="" type="checkbox"/>
vehicleModel <input checked="" type="checkbox"/>
price <input checked="" type="checkbox"/> fuelType
<input checked="" type="checkbox"/> timestamps
<input checked="" type="checkbox"/>

vehicleSchema (schema name)
-----------------------------



## QUESTION SET - 16

1. Create a schema-model according to the following diagrams (box shown below).
- 2.

type:

- type, mandatory, trim brand:
- type, mandatory, trim vehicleModel:
- type, mandatory, trim price:
- type, mandatory fuelType:
- type, mandatory

timestamps: →

created time

Perform **CREATE, EDIT, SHOW** functionality. (use forms wisely if needed)  
**‘/weather/new’, ‘/weather/:id/edit’, ‘/weather/:id’** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **locationName** 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.

Weather (model name)

## QUESTION SET - 17

1. Create a schema-model according to the following diagrams (box shown below).
- 2.

locationType ✓  
locationName ✓  
temperature ✓  
humidity ✓ windSpeed  
✓ precipitation ✓  
timestamps ✓

weatherSchema (schema name)

locationType:  
→ type, mandatory, trim locationName:  
→ type, mandatory, trim temperature:  
→ type, mandatory humidity:  
→ type, mandatory windSpeed:  
→ type, mandatory precipitation:  
→ type, mandatory  
timestamps: →  
created time

Perform **READ, CREATE, EDIT, SHOW** functionality. (use forms wisely if needed)

**‘/spaceships’, ‘/spaceships/new’, ‘/spaceships/:id, ‘/spaceships/:id/edit** (use proper routing convention)

3. Form should have inputs according to the table below.
4. # Only **spaceshipModel** 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

## QUESTION SET - 18

1. Create a schema-model according to the following diagrams (box shown below).
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
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.

Spaceship (model name)

category ✓  
spaceshipModel ✓  
manufacturer ✓  
fuelType ✓ maxSpeed  
✓ crewCapacity ✓  
timestamps ✓

spaceshipSchema (schema name)

category:  
→ type, mandatory, trim spaceshipModel:  
→ type, mandatory, trim manufacturer:  
→ type, mandatory, trim fuelType:  
→ type, mandatory, trim maxSpeed:  
→ type, mandatory  
crewCapacity: →  
type, mandatory  
timestamps: →  
created time

