3rd YEAR TEST-10 TOTAL TIME → 3hrs

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) % 10 = 291029063%10 = 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).
→ perform **Client-Side validation** as well (using Bootstrap)
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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login'** , **'/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13.No need to have any dummy data.
14.Do not share the node_modules and package-lock.json.

User (model name)	Song (model name)
Name ✅ Email ✅ Age ✅ isArtist ✅	songTitle ✅ singer✅ genre ✅ releaseDate ✅ songImageUrl✅

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

Question Set - 1

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)

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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login' , '/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13.No need to have any dummy data.
14.Do not share the node_modules and package-lock.json.

User(model name)	Contact (model name)
Name ✅ gender ✅ Age ✅ Aadhar card number ✅	contactName ✅ phoneNo✅ email✅ imageUrl✅ Timestamp ✅

userSchema(schema name)	contactSchema (schema name)
name : → type , mandatory , trim gender: → type , mandatory , trim age: → type , mandatory , trim Aadhar card: → type , mandatory	contactName : → type , mandatory , trim phoneNo: → type , mandatory , trim email: → type , mandatory , trim imageUrl: → type , mandatory Timestamp : → created time

Question Set - 2

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)
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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login' , '/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures

9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13. No need to have any dummy data.
14. Do not share the node_modules and package-lock.json.

User (model name)	RealEstate (model name)
Name ✅ Email ✅ Age ✅ Buyer / Seller ✅	propertyName ✅ owner✅ totalDimension ✅ Location ✅ imageUrl ✅

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

Question Set - 3

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)

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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login'** , **'/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13. No need to have any dummy data.
14. Do not share the node_modules and package-lock.json.

User(model name)	Event (model name)
Name ✅ Email ✅ Age ✅ ManagedBy ✅ Address ✅	name ✅ location ✅ date ✅ attendees ✅ Timestamp ✅

userSchema(schema name)	eventSchema (schema name)
name: → type , mandatory , trim Email: → type , mandatory , trim age: → type , mandatory , trim managedBy: → type , mandatory Address: → type , mandatory	name: → type , mandatory , trim Location: → type , mandatory , trim Date: → type , mandatory , trim attendees: → type , mandatory timestamps: → created time

Question Set - 4

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)
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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login' , '/logout** and show persons name on the top right next to login button.
7. Use <u>*session*</u> and <u>*passport*</u> to perform relevant tasks.

8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13.No need to have any dummy data.
14.Do not share the node_modules and package-lock.json.

User (model name)	Car (model name)
Name ✅ Email ✅ Age ✅ isOwner ✅ Address ✅	company✅ model✅ year✅ mileage✅ Timestamp ✅

userSchema (schema name)	carSchema (schema name)
name: → type , mandatory , trim email: → type , mandatory , trim age: → type , mandatory , trim isOwner: → type , mandatory Address: → type, mandatory	company: → type , mandatory , trim model: → type , mandatory , trim year: → type , mandatory , trim mileage: → type , mandatory Timestamp: → created time

Question Set - 5

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)

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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login' , '/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13. No need to have any dummy data.
14. Do not share the node_modules and package-lock.json.

User (model name)	Mobile (model name)
Name ✅ Email ✅ Age ✅ Aadhar card number ✅ Address ✅	brand✅ model✅ operatingSystem✅ storageCapacity✅ releaseDate✅

userSchema(schema name)	mobileSchema (schema name)
name: → type , mandatory , trim email: → type , mandatory , trim age: → type , mandatory , trim Aadhar card number: → type , mandatory Address: → type , mandatory	brand: → type , mandatory , trim model: → type , mandatory , trim operatingSystem: → type , mandatory , trim storageCapacity: → type , mandatory releaseDate: → type , mandatory , trim

Question Set - 6

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)
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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login' , '/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema

12. **Routes** → make different routing files
13.No need to have any dummy data.
14.Do not share the node_modules and package-lock.json.

User(model name)	University (model name)
Name ✅ Email ✅ Phone No. ✅ userType ✅ Address ✅	universityName ✅ affiliateBy✅ location✅ numberOfCourses✅ ImagesUrl✅

userSchema(schema name)	mobileSchema (schema name)

name :
→ type , mandatory , trim
 	
email:
→ type , mandatory , trim
 	
phoneNo:
→ type , mandatory , trim
 	
userType:
→ type , mandatory
 	
Address :
→ type , mandatory , trim
 	
Eg:
{
userType :student/teacher
}
	universityName :
	→ type , mandatory , trim
	
	affiliateBy:
	→ type , mandatory , trim
	
	location:
	→ type , mandatory , trim
	
	numberOfCourses:
	→ type , mandatory
	
	ImagesUrl:
	→ type , mandatory , trim

Question Set - 7

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)

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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login' , '/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13.No need to have any dummy data.
14.Do not share the node_modules and package-lock.json.

User (model name)	Restaurant (model name)
Name ✅ Email ✅ Phone No. ✅ Aadhar card number ✅ Address ✅	restaurantName ✅ Cuisine ✅ Chef ✅ phoneNo ✅ ImageUrl ✅

userSchema(schema name)	restaurantSchema (schema name)
name : → type , mandatory , trim email: → type , mandatory , trim phoneNo: → type , mandatory , trim Aadhar card number: → type , mandatory Address : → type , mandatory , trim	restaurantName : → type , mandatory , trim Cuisine : → type , mandatory , trim Chef : → type , mandatory , trim phoneNo : → type , mandatory ImageUrl : → type , mandatory , trim

Question Set - 8

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)

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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login'** , **'/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)

10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13. No need to have any dummy data.
14. Do not share the node_modules and package-lock.json.

User (model name)	Airline (model name)
Name ✅ age ✅ Phone No. ✅ gender ✅ Address ✅	companyName ✅ owner✅ numberOfPlane ✅ planeCapacity ✅ Model ✅

User (schema name)	airlineSchema (schema name)
name : → type , mandatory , trim age: → type , mandatory , trim phoneNo: → type , mandatory , trim gender: → type , mandatory Address : → type , mandatory , trim	companyName : → type , mandatory , trim owner: → type , mandatory , trim numberOfPlane : → type , mandatory , trim planeCapacity : → type , mandatory Model : → type , mandatory , trim

Question Set - 9

1. Create a schema-model according to the following diagrams (box shown below).
→ perform **Client-Side validation** as well (using Bootstrap)
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. Perform Authentication with passport. (following the schema structure)
6. **/register'** , **'/login'** , **'/logout** and show persons name on the top right next to login button.
7. Use *session* and *passport* to perform relevant tasks.
8. Follow proper folder structures
9. **Public** → style according to the template shown (just for reference)
10. **Views** → create proper template files
11. **Models** → make different schema
12. **Routes** → make different routing files
13.No need to have any dummy data.
14.Do not share the node_modules and package-lock.json.

User (model name)	Hotel (model name)
Name ✅ Email ✅ Phone No. ✅ userType ✅ Address ✅	hotelName ✅ rating ✅ numberOfRoom ✅ Location ✅ roomImageUrl ✅

userSchema(schema name)	hotelSchema (schema name)

name :
→ type , mandatory , trim
 	
email:
→ type , mandatory , trim
 	
phoneNo:
→ type , mandatory , trim
 	
userType:
→ type , mandatory
 	
Address :
→ type , mandatory , trim
 	
Eg:
{
    userType :employee/guest
}
	hotelName :
	→ type , mandatory , trim
	
	rating:
	→ type , mandatory , trim
	
	numberOfRoom :
	→ type , mandatory , trim
	
	Location :
	→ type , mandatory
	
	roomImageUrl :
	→ type , mandatory , trim