Backend 1.3_CW Exercises

Querying the Movie Model

Step 1: Create a Movie

Create a function `createMovie` that accepts an object containing movie data and adds a new movie to the database.

```
async function createCat() {
  try {
    const newCat = new Cat({
      name: 'Whiskers',
      age: 3,
      breed: 'Persian',
      color: 'White',
    })

    const savedCat = await newCat.save()
    console.log('New cat created:', savedCat)
  } catch (error) {
    console.error('Error creating cat:', error)
  }
}

createCat()
```

COPY

https://replit.com/@tanaypratap/BE13CW-ex01

- Solution

```
const newMovie = {
  title: 'New Movie',
  releaseYear: 2023,
  genre: ['Drama'],
  director: 'Director Name',
  actors: ['Actor 1', 'Actor 2'],
  language: 'Hindi',
  country: 'India',
  rating: 7.5,
  plot: 'Plot of the movie',
  awards: 'Awards received',
  posterUrl: '<https://example.com/poster.jpg>',
  trailerUrl: '<https://example.com/trailer.mp4>',
}

async function createMovie(movieData) {
  try {
    const movie = new Movie(movieData)
    const savedMovie = await movie.save()
    console.log('Created movie:', savedMovie)
  } catch (error) {
```

```
      throw error
    }
  }

  // Call the function with the newMovie object and log the result
  createMovie(newMovie)
```

COPY

Step 2: Read a Movie

Create a function `readMovie` that accepts the movie title and retrieves the movie details from the database.

```
async function getCatByName(catName) {
  try {
    const foundCat = await Cat.findOne({ name: catName })
    if (foundCat) {
      console.log('Found cat:', foundCat)
    } else {
      console.log('Cat not found')
    }
  } catch (error) {
    console.error('Error getting cat:', error)
  }
}

getCatByName('Whiskers')
```

COPY

- Solution

```
  async function readMovie(movieTitle) {
    try {
      const movie = await Movie.findOne({ title: movieTitle })
      console.log(movie)
    } catch (error) {
      throw error
    }
  }

  // Call the function with a movie title and log the result
  readMovie('Dilwale Dulhania Le Jayenge')
```

COPY

Step 3: Read All Movies

Create a function `readAllMovies` that retrieves all movies from the database.

```
async function getAllCats() {
  try {
    const allCats = await Cat.find({})
    console.log('All cats:', allCats)
  } catch (error) {
    console.error('Error getting cats:', error)
  }
}
```

```
getAllCats()
```

COPY

- Solution

```
async function readAllMovies() {
  try {
    const allMovies = await Movie.find()
    console.log('All movies:', allMovies)
  } catch (error) {
    throw error
  }
}

// Call the function and log the result
readAllMovies()
```

COPY

Step 4: Read All Movies for a Given Actor

Create a function `readMoviesByActor` that accepts an actor's name and retrieves all movies in which the actor has appeared.

```
async function getCatsByBreed(breed) {
  try {
    const catsOfBreed = await Cat.find({ breed: breed })
    console.log(`Cats of breed "${breed}":`, catsOfBreed)
  } catch (error) {
    console.error('Error getting cats:', error)
  }
}

getCatsByBreed('Persian')
```

COPY

- Solution https://replit.com/@tanaypratap/BE13CW-ex04

```
async function readMoviesByActor(actorName) {
  try {
    const moviesByActor = await Movie.find({ actors: actorName })
    console.log('Movies by actor:', moviesByActor)
  } catch (error) {
    throw error
  }
}

// Call the function with an actor's name and log the result
readMoviesByActor('Shah Rukh Khan')
```

COPY

Step 5: Read All Movies for a Given Director

Create a function `readMoviesByDirector` that accepts a director's name and retrieves all movies directed by that director.

- Solution

```
async function readMoviesByDirector(directorName) {
  try {
    const moviesByDirector = await Movie.find({ director: directorName })
    console.log('Movies by director:', moviesByDirector)
  } catch (error) {
    throw error
  }
}

// Call the function with a director's name and log the result
readMoviesByDirector('Rajkumar Hirani')
```

COPY

Step 6: Read All Movies for a Given Year

Create a function `readMoviesByYear` that accepts a release year and retrieves all movies released in that year.

- Solution

```
async function readMoviesByYear(year) {
  try {
    const moviesByYear = await Movie.find({ releaseYear: year })
    console.log('Movies by year:', moviesByYear)
  } catch (error) {
    throw error
  }
}

// Call the function with a release year and log the result
readMoviesByYear(2015)
```

COPY

Step 7: Read All Movies for a Given Genre

Create a function `readMoviesByGenre` that accepts a genre and retrieves all movies belonging to that genre.

- Solution https://replit.com/@tanaypratap/BE13CW-ex07

```
async function readMoviesByGenre(genre) {
  try {
    const moviesByGenre = await Movie.find({ genre: genre })
    console.log('Movies by genre:', moviesByGenre)
  } catch (error) {
    throw error
  }
}

// Call the function with a genre and log the result
readMoviesByGenre('Comedy')
```

COPY

Step 8: Update a Movie by ID

Create a function `updateMovie` that accepts a movie ID and an object with updated data, and updates the movie with the provided ID.

```
async function updateCatById(catId, updateData) {
  try {
    const updatedCat = await Cat.findByIdAndUpdate(catId, updateData, {
      new: true,
    })
    if (updatedCat) {
      console.log('Updated cat:', updatedCat)
    } else {
      console.log('Cat not found')
    }
  } catch (error) {
    console.error('Error updating cat:', error)
  }
}

// Example usage:
updateCatById('your-cat-id', { age: 4 })
```

COPY

- Solution

```
  async function updateMovie(movieId, updatedData) {
    try {
      const updatedMovie = await Movie.findByIdAndUpdate(movieId, updatedData, {
        new: true,
      })
      // Movie.findOneAndUpdate({ title: title }, updatedData, { new:true })
      console.log('Updated movie:', updatedMovie)
    } catch (error) {
      throw error
    }
  }

  // Call the function with a movie ID, updated data, and log the result
  updateMovie('your-movie-id-here', { rating: 8.5 })
```

```
const selectedMovie = await Movie.findById(movieId)
selectedMovie.rating = updatedRating
selectedMovie.save()
```

COPY

Step 9: Delete a Movie by ID

Create a function `deleteMovie` that accepts a movie ID and deletes the movie with the provided ID.

```
async function deleteCatById(catId) {
  try {
    const deletedCat = await Cat.findByIdAndDelete(catId)
    if (deletedCat) {
      console.log('Deleted cat:', deletedCat)
    } else {
      console.log('Cat not found')
    }
  } catch (error) {
    console.error('Error deleting cat:', error)
  }
}

// Example usage:
```

```
deleteCatById('your-cat-id')
```

COPY

- Solution

```
async function deleteMovie(movieId) {
  try {
    const deletedMovie = await Movie.findByIdAndDelete(movieId)
    console.log('Deleted movie:', deletedMovie)
  } catch (error) {
    throw error
  }
}

// Call the function with a movie ID and log the result
deleteMovie('your-movie-id-here')
```

COPY

Step 10: Read All Movies Sorted by Rating

Create a function `readMoviesByRating` that retrieves all movies from the database and sorts them in descending order based on their ratings.

```
async function getAllCatsSortedByAge() {
  try {
    const sortedCats = await Cat.find({}).sort({ age: 1 })
    console.log('All cats sorted by age:', sortedCats)
  } catch (error) {
    console.error('Error getting cats:', error)
  }
}

getAllCatsSortedByAge()
```

COPY

- Solution https://replit.com/@tanaypratap/BE13CW-ex10

```
async function readMoviesByRating() {
  try {
    const moviesByRating = await Movie.find().sort({ rating: -1 })
    console.log('Movies sorted by rating:', moviesByRating)
  } catch (error) {
    throw error
  }
}

// Call the function and log the result
readMoviesByRating()
```

COPY

Step 11: Read All Movies Sorted by Release Year

Create a function `readMoviesByReleaseYear` that retrieves all movies from the database and sorts them in ascending order based on their release years.

- Solution

```
async function readMoviesByReleaseYear() {
  try {
    const moviesByReleaseYear = await Movie.find().sort({ releaseYear: 1 })
    console.log('Movies sorted by release year:', moviesByReleaseYear)
  } catch (error) {
    throw error
  }
}

// Call the function and log the result
readMoviesByReleaseYear()
```