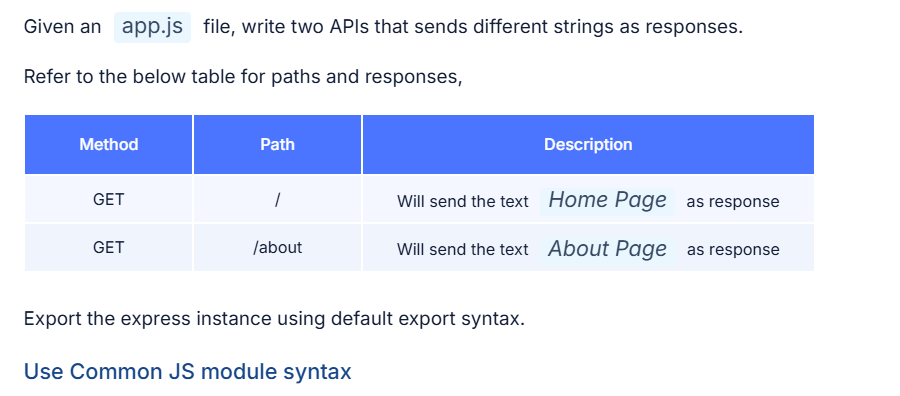
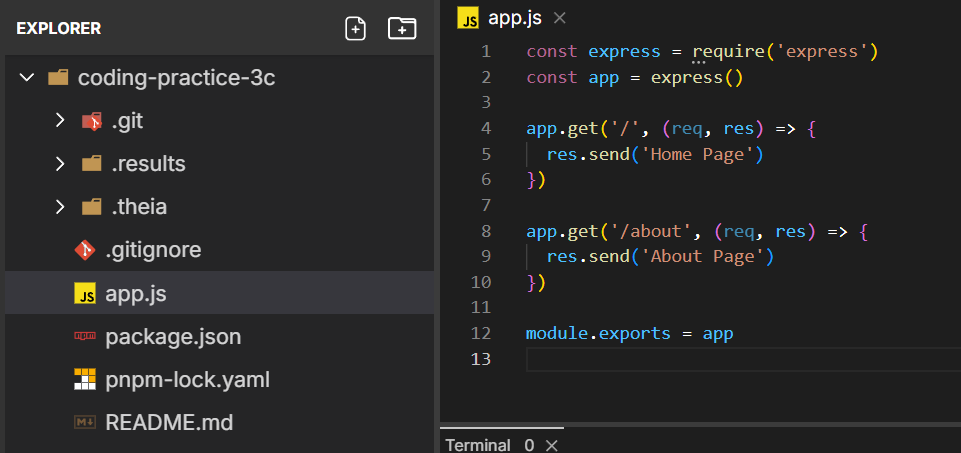
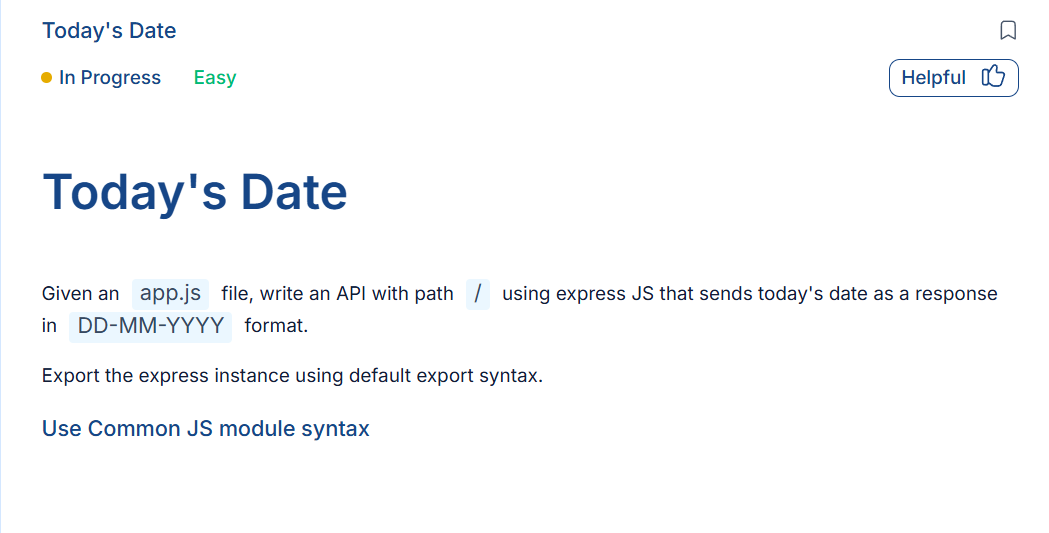
# **API Routing**

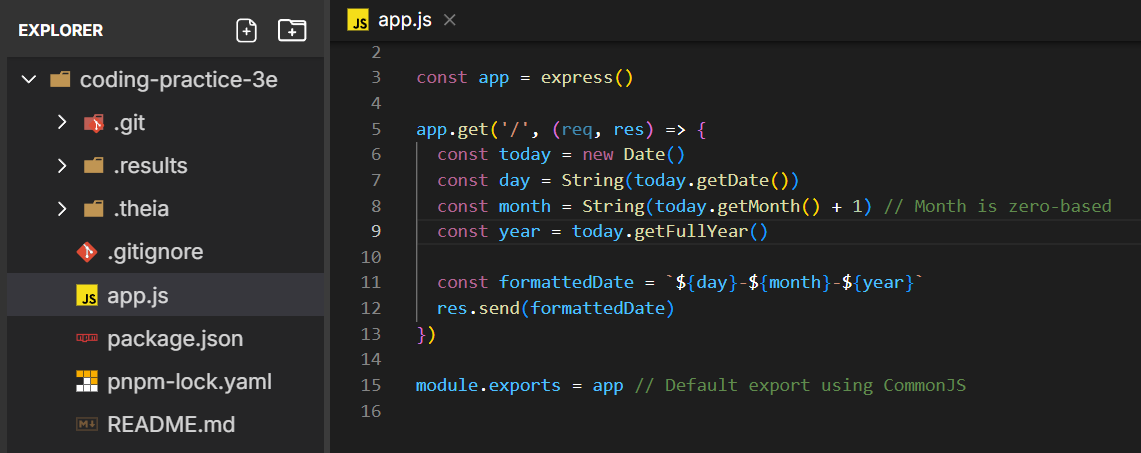


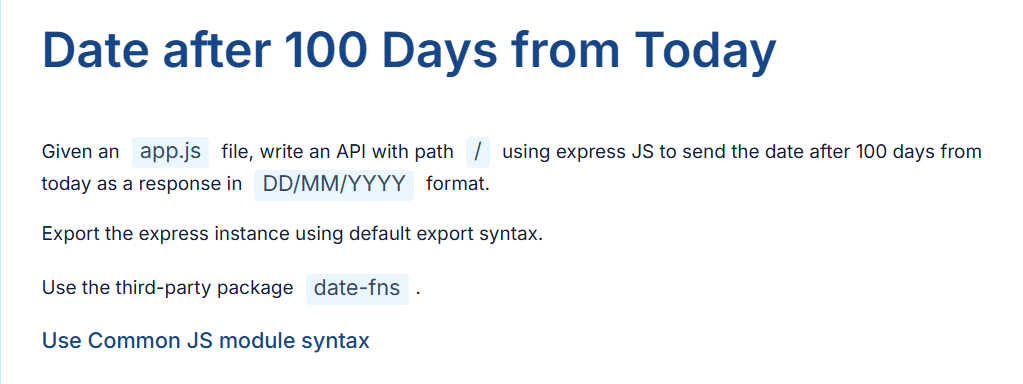
**Answer:**

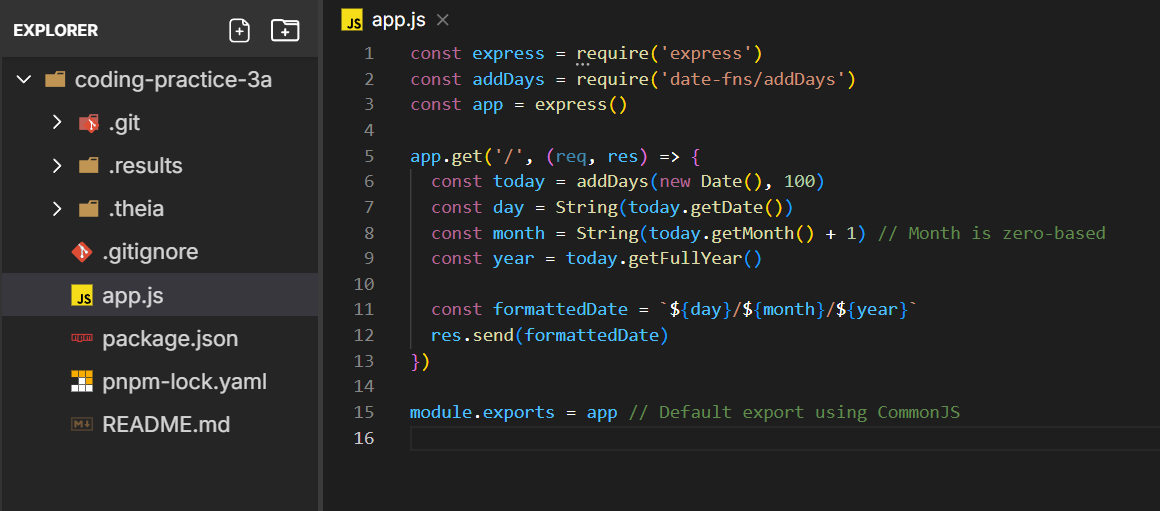


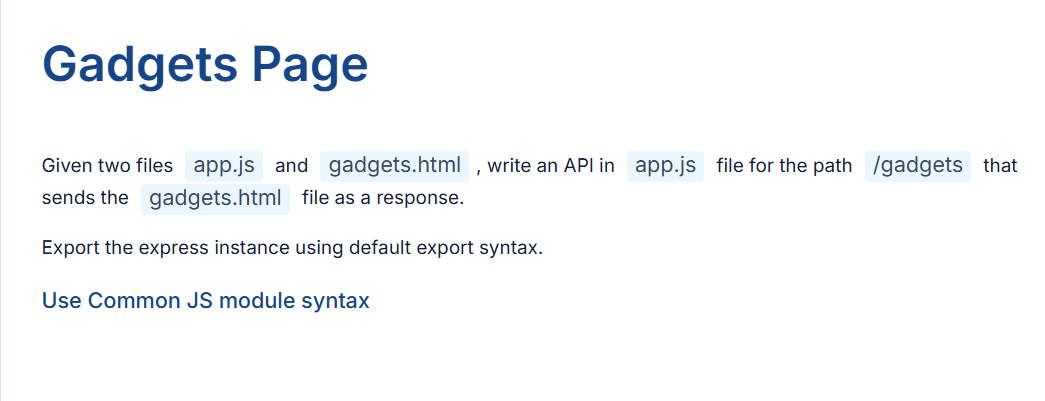


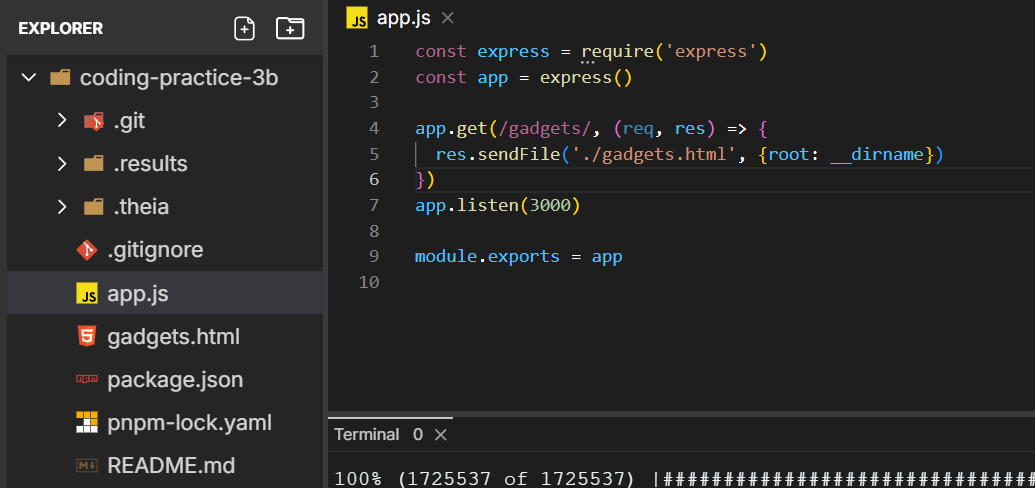
**Answer:**

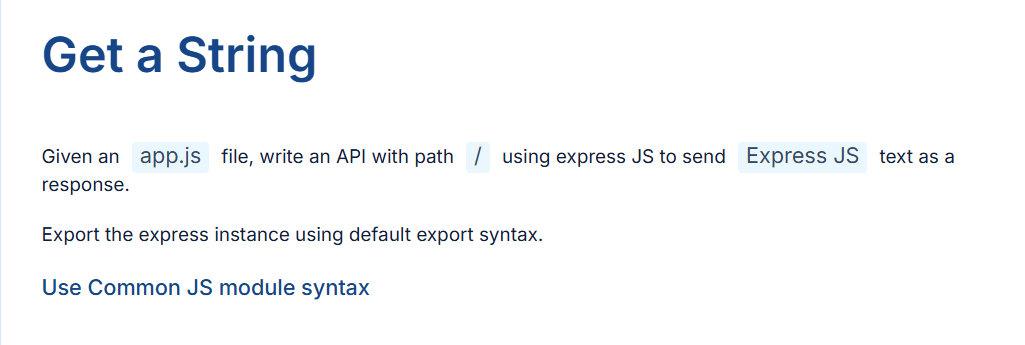


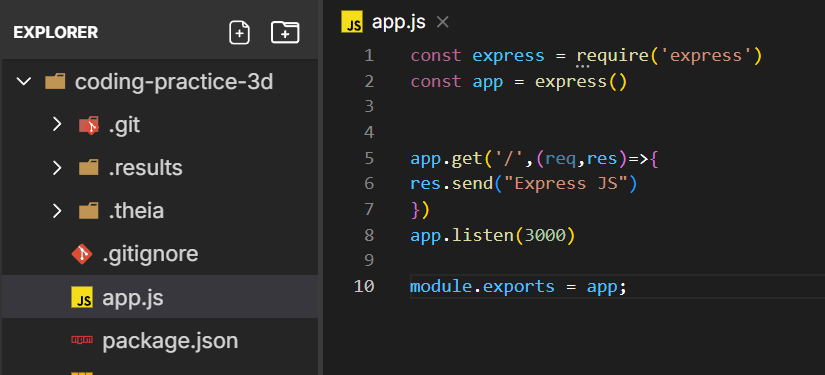


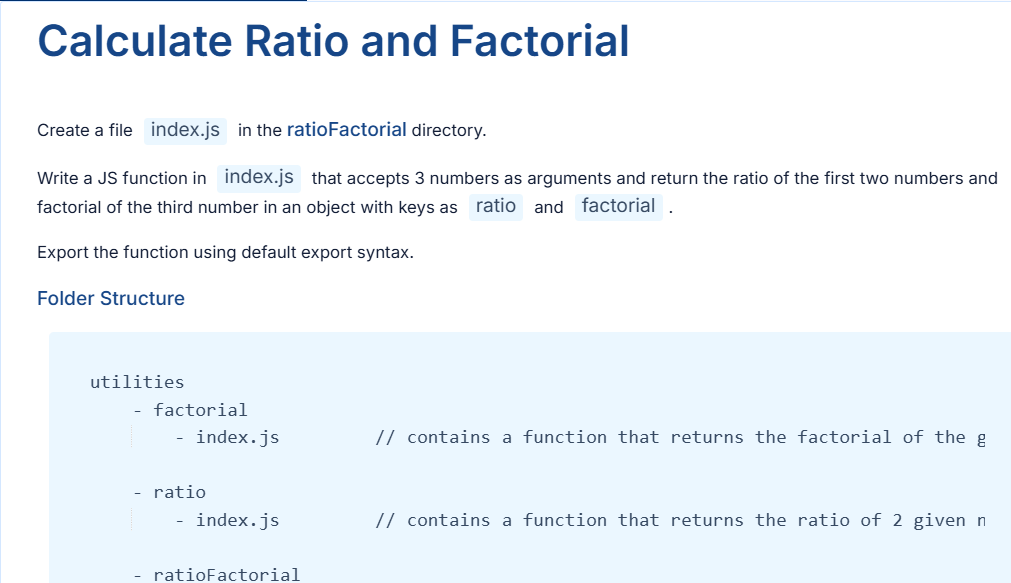


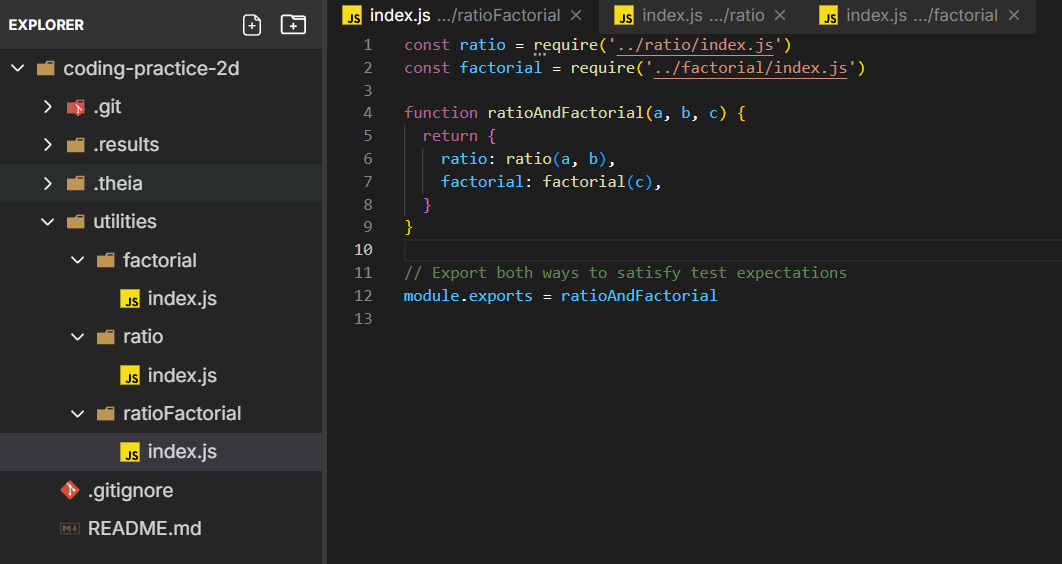




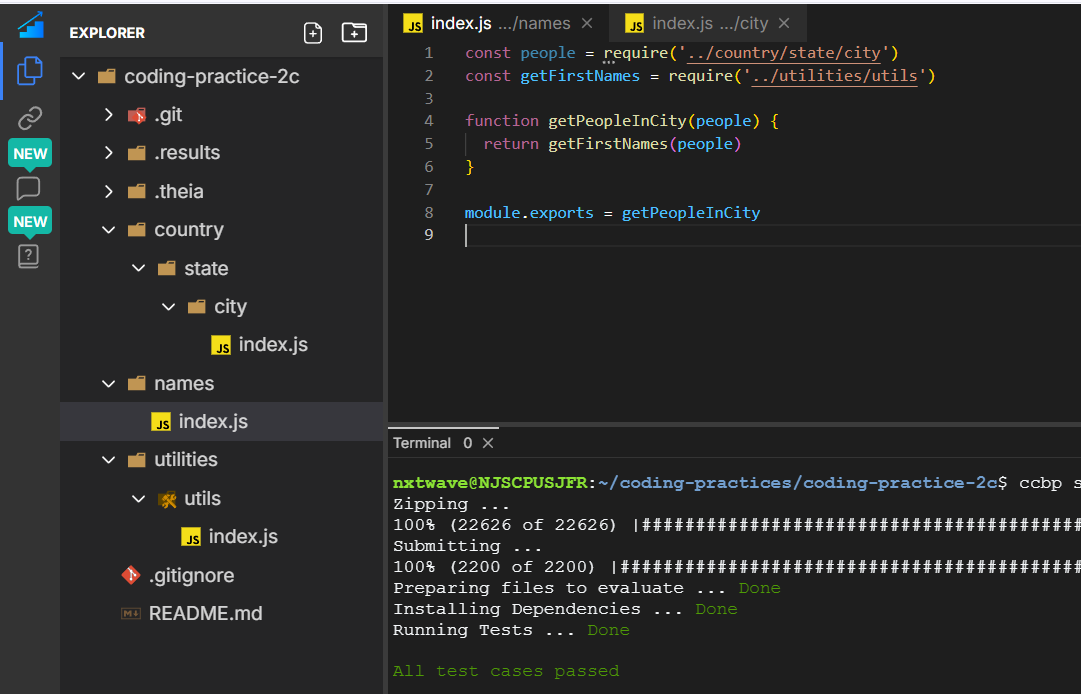


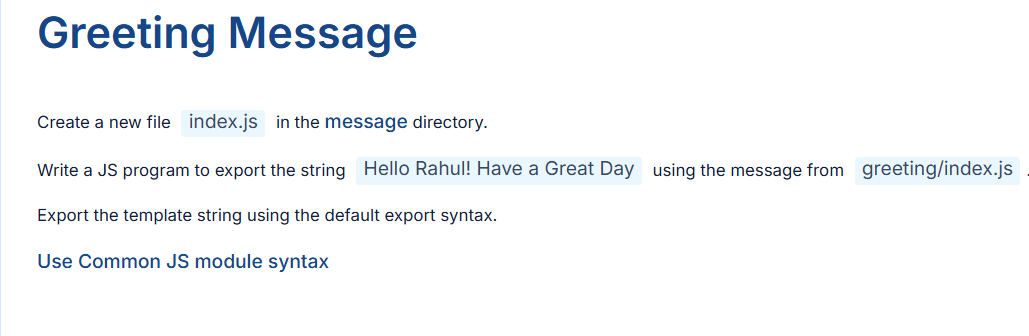


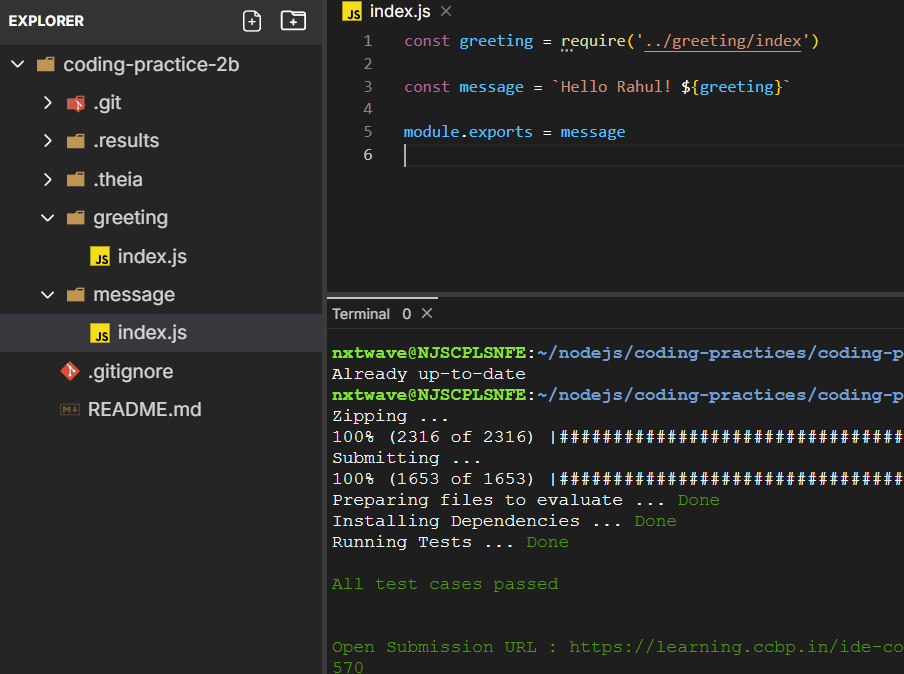


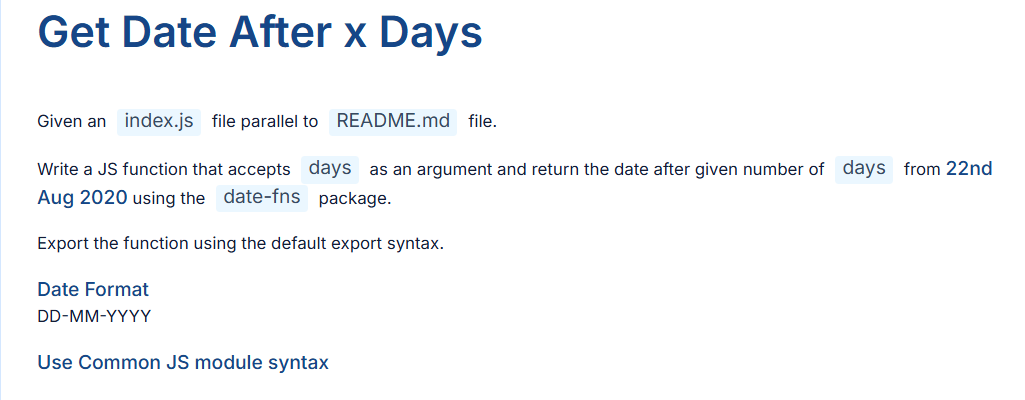


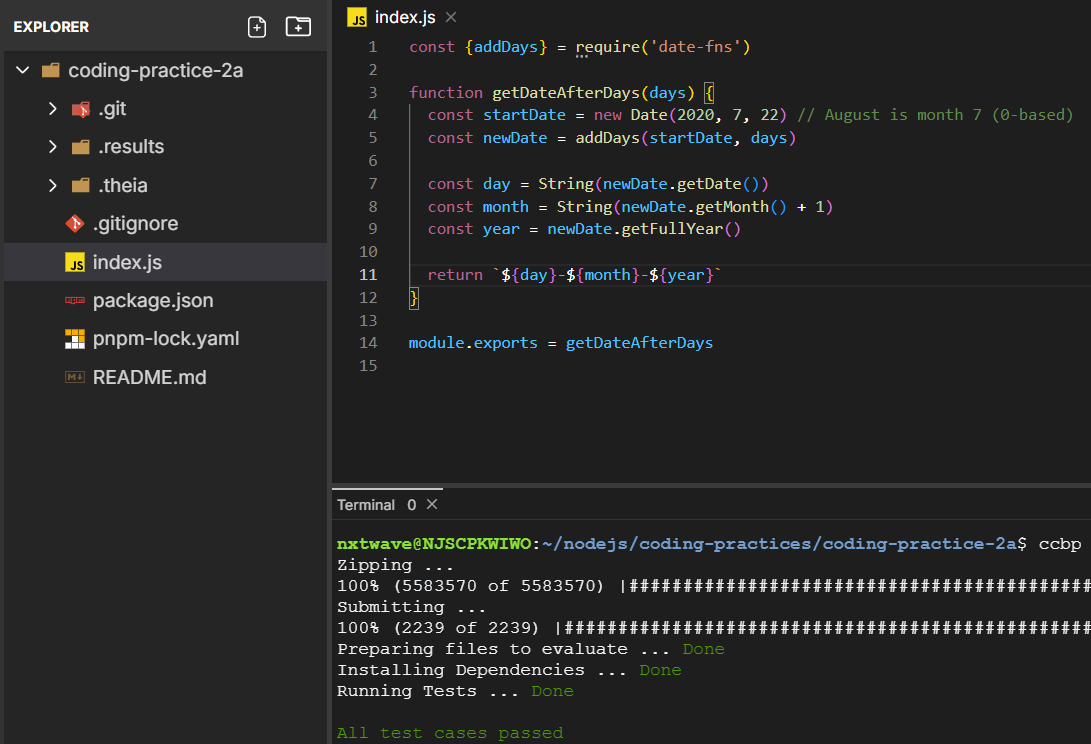




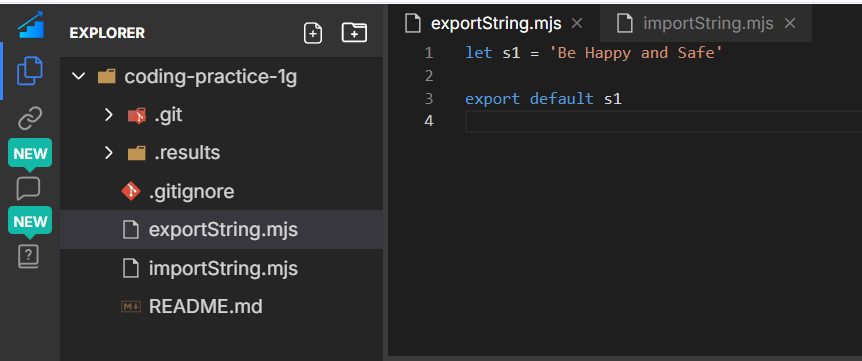


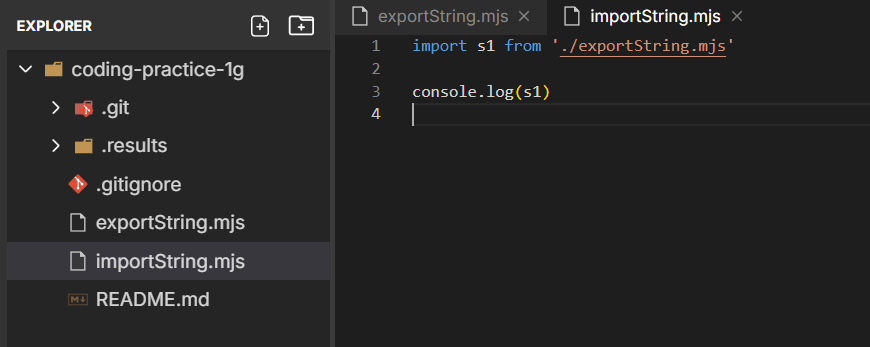














**Answer**

const express = require('express')

const sqlite3 = require('sqlite3')

const path = require('path')

const {open} = require('sqlite') // ✅ Import open from sqlite

const app = express()

app.use(express.json())

const dbPath = path.join(\_\_dirname, 'cricketTeam.db')

let db = null

const initializeDbAndServer = async () => {

try {

db = await open({

filename: dbPath,

driver: sqlite3.Database,

})

// ✅ Start server

app.listen(3000, () => {

console.log('Server Running at http://localhost:3000/')

})

} catch (e) {

console.log(`DB Error: ${e.message}`)

process.exit(1)

}

}

initializeDbAndServer()

// API 1: GET /players/ - Get all players

app.get('/players/', async (req, res) => {

try {

const query = `SELECT \* FROM cricket\_team;`

const rows = await db.all(query) // ✅ no callback

const formattedPlayers = rows.map(player => ({

playerId: player.player\_id,

playerName: player.player\_name,

jerseyNumber: player.jersey\_number,

role: player.role,

}))

res.json(formattedPlayers)

} catch (error) {

res.status(500).send('Database error')

}

})

// API 2: POST /players/ - Create new player

app.post('/players/', async (req, res) => {

const {playerName, jerseyNumber, role} = req.body

const addPlayerQuery = `

INSERT INTO

cricket\_team (player\_name, jersey\_number, role)

VALUES

('${playerName}', ${jerseyNumber}, '${role}');

`

await db.run(addPlayerQuery)

res.send('Player Added to Team')

})

// API 3: GET /players/:playerId/ - Get player by ID

app.get('/players/:playerId/', async (req, res) => {

const {playerId} = req.params

const getPlayerQuery = `

SELECT

player\_id AS playerId,

player\_name AS playerName,

jersey\_number AS jerseyNumber,

role

FROM

cricket\_team

WHERE

player\_id = ${playerId};

`

const player = await db.get(getPlayerQuery)

res.send(player)

})

// API 4: PUT /players/:playerId/ - Update player details

app.put('/players/:playerId/', async (req, res) => {

const {playerId} = req.params

const {playerName, jerseyNumber, role} = req.body

const updatePlayerQuery = `

UPDATE cricket\_team

SET

player\_name = '${playerName}',

jersey\_number = ${jerseyNumber},

role = '${role}'

WHERE

player\_id = ${playerId};

`

await db.run(updatePlayerQuery)

res.send('Player Details Updated')

})

// API 5: DELETE /players/:playerId/ - Delete player

app.delete('/players/:playerId/', async (req, res) => {

const {playerId} = req.params

const deletePlayerQuery = `

DELETE FROM cricket\_team

WHERE player\_id = ${playerId};

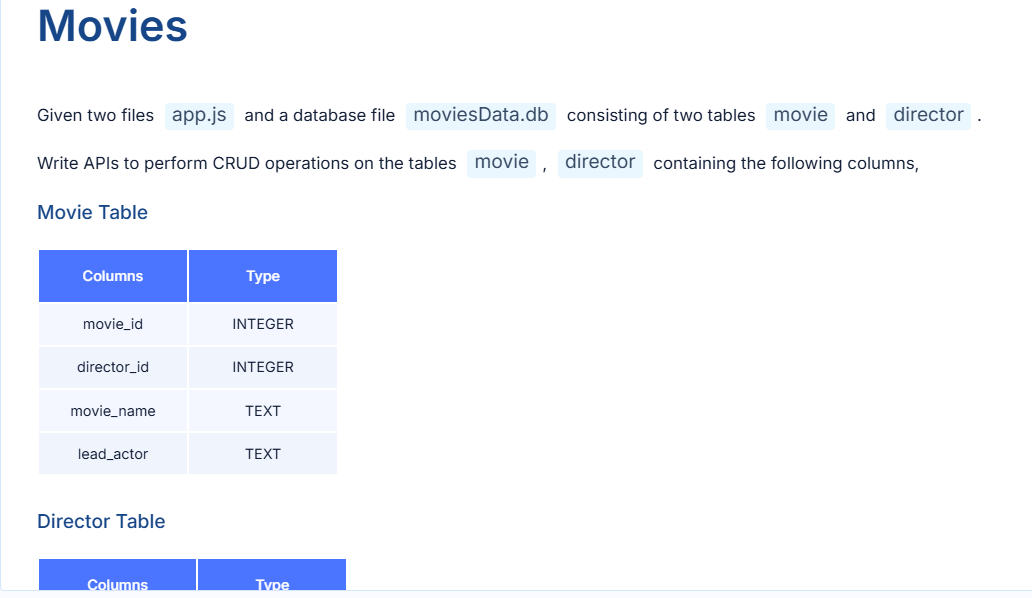
`

await db.run(deletePlayerQuery)

res.send('Player Removed')

})

module.exports = app



**Answer**

const express = require('express')

const sqlite3 = require('sqlite3')

const {open} = require('sqlite')

const path = require('path')

const app = express()

app.use(express.json())

const dbPath = path.join(\_\_dirname, 'moviesData.db')

let db = null

const initializeDbAndServer = async () => {

try {

db = await open({

filename: dbPath,

driver: sqlite3.Database,

})

app.listen(3000, () => {

console.log('Server Running at http://localhost:3000/')

})

} catch (error) {

console.log(`DB Error: ${error.message}`)

process.exit(1)

}

}

initializeDbAndServer()

// API 1: Get all movie names

app.get('/movies/', async (req, res) => {

const getMoviesQuery = `SELECT movie\_name FROM movie;`

const movies = await db.all(getMoviesQuery)

res.send(

movies.map(movie => ({

movieName: movie.movie\_name,

})),

)

})

// API 2: Add a new movie

app.post('/movies/', async (req, res) => {

const {directorId, movieName, leadActor} = req.body

const addMovieQuery = `

INSERT INTO movie (director\_id, movie\_name, lead\_actor)

VALUES (${directorId}, '${movieName}', '${leadActor}');

`

await db.run(addMovieQuery)

res.send('Movie Successfully Added')

})

// API 3: Get movie by ID

app.get('/movies/:movieId/', async (req, res) => {

const {movieId} = req.params

const getMovieQuery = `

SELECT \* FROM movie WHERE movie\_id = ${movieId};

`

const movie = await db.get(getMovieQuery)

res.send({

movieId: movie.movie\_id,

directorId: movie.director\_id,

movieName: movie.movie\_name,

leadActor: movie.lead\_actor,

})

})

// API 4: Update movie by ID

app.put('/movies/:movieId/', async (req, res) => {

const {movieId} = req.params

const {directorId, movieName, leadActor} = req.body

const updateQuery = `

UPDATE movie

SET

director\_id = ${directorId},

movie\_name = '${movieName}',

lead\_actor = '${leadActor}'

WHERE movie\_id = ${movieId};

`

await db.run(updateQuery)

res.send('Movie Details Updated')

})

// API 5: Delete movie by ID

app.delete('/movies/:movieId/', async (req, res) => {

const {movieId} = req.params

const deleteQuery = `

DELETE FROM movie WHERE movie\_id = ${movieId};

`

await db.run(deleteQuery)

res.send('Movie Removed')

})

// API 6: Get all directors

app.get('/directors/', async (req, res) => {

const getDirectorsQuery = `

SELECT \* FROM director;

`

const directors = await db.all(getDirectorsQuery)

res.send(

directors.map(director => ({

directorId: director.director\_id,

directorName: director.director\_name,

})),

)

})

// API 7: Get all movies by a director

app.get('/directors/:directorId/movies/', async (req, res) => {

const {directorId} = req.params

const getMoviesQuery = `

SELECT movie\_name FROM movie WHERE director\_id = ${directorId};

`

const movies = await db.all(getMoviesQuery)

res.send(

movies.map(movie => ({

movieName: movie.movie\_name,

})),

)

})

module.exports = app



**Answer**

const express = require('express')

const sqlite3 = require('sqlite3')

const {open} = require('sqlite')

const path = require('path')

const app = express()

app.use(express.json())

const dbPath = path.join(\_\_dirname, 'covid19India.db')

let db = null

const initializeDbAndServer = async () => {

try {

db = await open({

filename: dbPath,

driver: sqlite3.Database,

})

app.listen(3000, () => {

console.log('Server running at http://localhost:3000/')

})

} catch (error) {

console.error(`DB Error: ${error.message}`)

process.exit(1)

}

}

initializeDbAndServer()

// API 1: Get all states

app.get('/states/', async (req, res) => {

const query = `SELECT \* FROM state;`

const states = await db.all(query)

res.send(

states.map(state => ({

stateId: state.state\_id,

stateName: state.state\_name,

population: state.population,

})),

)

})

// API 2: Get state by ID

app.get('/states/:stateId/', async (req, res) => {

const {stateId} = req.params

const query = `SELECT \* FROM state WHERE state\_id = ${stateId};`

const state = await db.get(query)

res.send({

stateId: state.state\_id,

stateName: state.state\_name,

population: state.population,

})

})

// API 3: Create district

app.post('/districts/', async (req, res) => {

const {districtName, stateId, cases, cured, active, deaths} = req.body

const query = `

INSERT INTO district (district\_name, state\_id, cases, cured, active, deaths)

VALUES (

'${districtName}', ${stateId}, ${cases}, ${cured}, ${active}, ${deaths}

);

`

await db.run(query)

res.send('District Successfully Added')

})

// API 4: Get district by ID

app.get('/districts/:districtId/', async (req, res) => {

const {districtId} = req.params

const query = `SELECT \* FROM district WHERE district\_id = ${districtId};`

const district = await db.get(query)

res.send({

districtId: district.district\_id,

districtName: district.district\_name,

stateId: district.state\_id,

cases: district.cases,

cured: district.cured,

active: district.active,

deaths: district.deaths,

})

})

// API 5: Delete district by ID

app.delete('/districts/:districtId/', async (req, res) => {

const {districtId} = req.params

const query = `DELETE FROM district WHERE district\_id = ${districtId};`

await db.run(query)

res.send('District Removed')

})

// API 6: Update district by ID

app.put('/districts/:districtId/', async (req, res) => {

const {districtId} = req.params

const {districtName, stateId, cases, cured, active, deaths} = req.body

const query = `

UPDATE district

SET

district\_name = '${districtName}',

state\_id = ${stateId},

cases = ${cases},

cured = ${cured},

active = ${active},

deaths = ${deaths}

WHERE district\_id = ${districtId};

`

await db.run(query)

res.send('District Details Updated')

})

// API 7: Get state stats by state ID

app.get('/states/:stateId/stats/', async (req, res) => {

const {stateId} = req.params

const query = `

SELECT

SUM(cases) as totalCases,

SUM(cured) as totalCured,

SUM(active) as totalActive,

SUM(deaths) as totalDeaths

FROM district

WHERE state\_id = ${stateId};

`

const stats = await db.get(query)

res.send({

totalCases: stats.totalCases,

totalCured: stats.totalCured,

totalActive: stats.totalActive,

totalDeaths: stats.totalDeaths,

})

})

// API 8: Get state name of a district by district ID

app.get('/districts/:districtId/details/', async (req, res) => {

const {districtId} = req.params

const query = `

SELECT state.state\_name as stateName

FROM district

INNER JOIN state ON district.state\_id = state.state\_id

WHERE district.district\_id = ${districtId};

`

const result = await db.get(query)

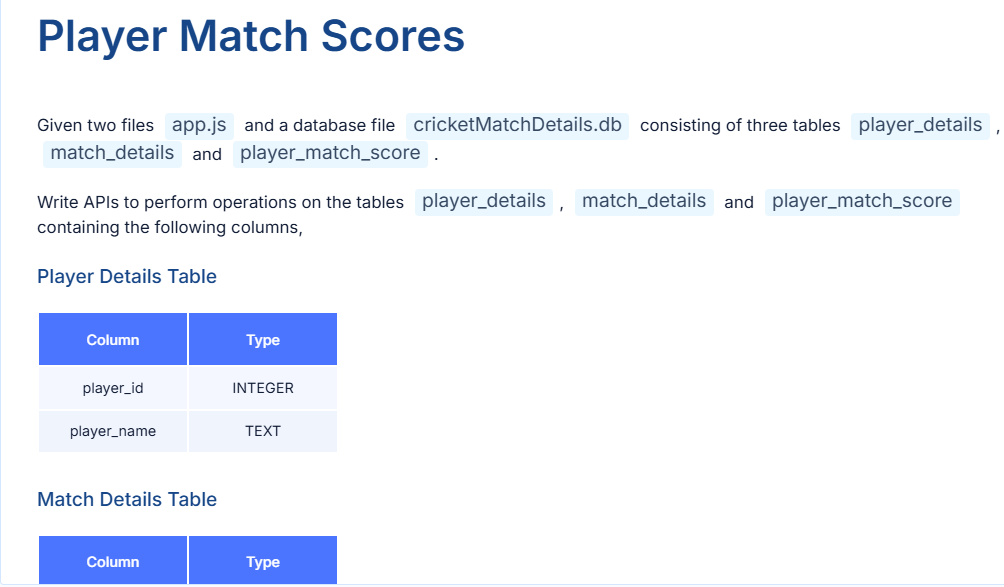
res.send({

stateName: result.stateName,

})

})

module.exports = app



**Answer**

const express = require('express')

const sqlite3 = require('sqlite3')

const {open} = require('sqlite')

const path = require('path')

const app = express()

app.use(express.json())

const dbPath = path.join(\_\_dirname, 'cricketMatchDetails.db')

let db = null

const initializeDbAndServer = async () => {

try {

db = await open({

filename: dbPath,

driver: sqlite3.Database,

})

app.listen(3000, () => {

console.log('Server running at http://localhost:3000/')

})

} catch (e) {

console.error(`DB Error: ${e.message}`)

process.exit(1)

}

}

initializeDbAndServer()

// API 1: GET /players/ - list all players

app.get('/players/', async (req, res) => {

const query = `SELECT \* FROM player\_details;`

const players = await db.all(query)

res.send(

players.map(player => ({

playerId: player.player\_id,

playerName: player.player\_name,

})),

)

})

// API 2: GET /players/:playerId/ - get player by ID

app.get('/players/:playerId/', async (req, res) => {

const {playerId} = req.params

const query = `SELECT \* FROM player\_details WHERE player\_id = ${playerId};`

const player = await db.get(query)

if (player) {

res.send({

playerId: player.player\_id,

playerName: player.player\_name,

})

} else {

res.status(404).send('Player Not Found')

}

})

// API 3: PUT /players/:playerId/ - update player name

app.put('/players/:playerId/', async (req, res) => {

const {playerId} = req.params

const {playerName} = req.body

const query = `

UPDATE player\_details

SET player\_name = '${playerName}'

WHERE player\_id = ${playerId};

`

await db.run(query)

res.send('Player Details Updated')

})

// API 4: GET /matches/:matchId/ - get match details

app.get('/matches/:matchId/', async (req, res) => {

const {matchId} = req.params

const query = `SELECT \* FROM match\_details WHERE match\_id = ${matchId};`

const match = await db.get(query)

if (match) {

res.send({

matchId: match.match\_id,

match: match.match,

year: match.year,

})

} else {

res.status(404).send('Match Not Found')

}

})

// API 5: GET /players/:playerId/matches - get all matches of a player

app.get('/players/:playerId/matches', async (req, res) => {

const {playerId} = req.params

const query = `

SELECT match\_details.match\_id as matchId, match\_details.match, match\_details.year

FROM player\_match\_score

INNER JOIN match\_details ON player\_match\_score.match\_id = match\_details.match\_id

WHERE player\_match\_score.player\_id = ${playerId};

`

const matches = await db.all(query)

res.send(matches)

})

// API 6: GET /matches/:matchId/players - get all players in a match

app.get('/matches/:matchId/players', async (req, res) => {

const {matchId} = req.params

const query = `

SELECT player\_details.player\_id as playerId, player\_details.player\_name as playerName

FROM player\_match\_score

INNER JOIN player\_details ON player\_match\_score.player\_id = player\_details.player\_id

WHERE player\_match\_score.match\_id = ${matchId};

`

const players = await db.all(query)

res.send(players)

})

// API 7: GET /players/:playerId/playerScores - get player stats (total score, fours, sixes)

app.get('/players/:playerId/playerScores', async (req, res) => {

const {playerId} = req.params

const query = `

SELECT

player\_details.player\_id as playerId,

player\_details.player\_name as playerName,

SUM(player\_match\_score.score) as totalScore,

SUM(player\_match\_score.fours) as totalFours,

SUM(player\_match\_score.sixes) as totalSixes

FROM player\_match\_score

INNER JOIN player\_details ON player\_match\_score.player\_id = player\_details.player\_id

WHERE player\_match\_score.player\_id = ${playerId}

GROUP BY player\_details.player\_id;

`

const stats = await db.get(query)

if (stats) {

res.send(stats)

} else {

res.status(404).send('Player Not Found or No Scores')

}

})

module.exports = app