Unit testing of the backend using JEST and Super test

Tested by: Vyom Patel

Unit testing is performed by using the following type of test cases:

- **Positive test cases:** These test cases verify that the system behaves correctly when given valid input. Where we had tested to see if a function returns the expected result when passed valid input parameters.

- **Negative test cases:** These cases verify if the system handles invalid input correctly. Tested to see if a function throws an error when passing invalid input parameters.

- **Boundary test cases:** These test cases verify that the system handles boundary cases correctly. Tested to see that a function handles the smallest and largest possible input values correctly.

- **Error handling test cases:** These test cases verify that the system handles errors and exceptions correctly. Tested to see that a function throws the correct error message when a particular error condition occurs.

User sign-up and log-in testing

(i) For successful sign-up

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/user_signup', () => {
    test('should return status 200 for Successful signup', async () => {
        const req = {
            body: {
                username : 'Vyom',
                email: '123@gmail.com',
                password: 'Vvp1_',
                phone_no : '1234567890',
                gender: "male"
            }
        };
        const res = await request(baseURL).post('/api/user_signup').send(req.body);
        expect(res.status).toEqual(200);
    });
```

(ii) sign up for users with different details but the same username.

```
test('should return status 400 for Username already exist', async () => {
    const req = {
        body: {
            username : 'Vyom',
            email: 'abc@gmail.com',
            password: 'Vvp1_',
            phone_no : '1234567890',
            gender: "male"
        }
    };
    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

iii) For Valid Email-Address:

```
test('should return status 400 for Enter Valid Email-Address', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123@gmailcom',
            password: '123',
            phone_no : '1234567890',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter Valid Email-Address', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123gmail.com',
            password: '123',
            phone_no : '1234567890',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

iv) For valid Password:

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123@gmail.com',
            password: '123',
            phone_no : '1234567890',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123@gmail.com',
            password: '123Aa',
            phone_no : '1234567890',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123@gmail.com',
            password: '123Aa',
            phone_no : '1234567890',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

v)For checking Phone Number format.

```
test('should return status 400 for Enter 10 Digit Phone-Number', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123@gmail.com',
            password: '123Aa_',
            phone_no : '123456789',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter 10 Digit Phone-Number', async () => {
    const req = {
        body: {
            username : 'meet',
            email: '123@gmail.com',
            password: '123Aa_',
            phone_no : '123456789m',
            gender: "male"
        }
    };

    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

vi) Checking for server error.

```
test('should return status 500 for Internal Server Error', async () => {
    const req = {
        body: {
            username : '',
            email: '123@gmail.com',
            password: '123Aa_',
            phone_no : '1234567890',
            gender: "male"
        }
    };
    const res = await request(baseURL).post('/api/user_signup').send(req.body);
    expect(res.status).toEqual(500);
});
```

vii) Tests for valid credentials.

```
describe('POST /api/user_login', () => {
    test('should return status 200 for valid credentials', async () => {
        const req = {
            body: {
                username: 'Vyom',
                password: 'Vvp1_'
            }
        };
        const res = await request(baseURL).post('/api/user_login').send(req.body);
        expect(res.status).toEqual(200);
    });
```

```
 test('should return status 400 for User not found', async () => {
     const req = {
         body: {
             username: 'Vvvpc',
             password: '123'
         }
     };
     const res = await request(baseURL).post('/api/user_login').send(req.body);
     expect(res.status).toEqual(400);
 });
```

viii) Tests for Invalid password.

```
test('should return status 400 for Invalid Password', async () => {
    const req = {
        body: {
            username: 'Vyom',
            password: '1234'
        }
    };
    const res = await request(baseURL).post('/api/user_login').send(req.body);
    expect(res.status).toEqual(400);
});
```

Results of the above test cases:

```
 PASS  test/user.test.js
  POST /api/user_signup
    √ should return status 200 for Successful signup (1303 ms)
    √ should return status 400 for Username already exist (36 ms)
    √ should return status 400 for Enter Valid Email-Address (32 ms)
    √ should return status 400 for Enter Valid Email-Address (30 ms)
    √ should return status 400 for Enter Valid Password (39 ms)
    √ should return status 400 for Enter Valid Password (28 ms)
    √ should return status 400 for Enter Valid Password (30 ms)
    √ should return status 400 for Enter 10 Digit Phone-Number (27 ms)
    √ should return status 400 for Enter 10 Digit Phone-Number (32 ms)
    √ should return status 500 for Internal Server Error (46 ms)
  POST /api/user_login
    √ should return status 200 for valid credentials (558 ms)
    √ should return status 400 for User not found (39 ms)
    √ should return status 400 for Invalid Password (354 ms)

Test Suites: 1 passed, 1 total
Tests:       13 passed, 13 total
Snapshots:   0 total
Time:        3.47 s
Ran all test suites.
PS C:\Users\Vyom Patel\Desktop\proj\backend>
```

Log in and sign up of organizer

i)Test for Successful login.

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/organizer_signup', () => {
    test('should return status 200 for Successful signup', async () => {
        const req = {
            body: {
                    "username":"Vyomxy",
                    "email" : "abc@gmail.com",
                    "password" : "A1_a",
                    "track_list":[
                        {
                            "track_name": "new_track",
                            "start_date": "2023-04-22T20:09:27.848+00:00",
                            "end_date": "2023-04-22T20:09:27.848+00:00"
                        }
                    ],
                    "resume_link": "abc"
            }
        };
        const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
        expect(res.status).toEqual(200);
    });
```

ii)Test for checking Username.

```
test('should return status 300 for Username already exists', async () => {
    const req = {
        body: {
            "username":"Vyomxy",
            "email" : "abc@gmail.com",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(300);
});
```

iii)Test for checking Valid Email address.

```
test('should return status 400 for enter valid email address', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmailcom",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for enter valid email address', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abcgmail.com",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for enter valid email address', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmail@com",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for enter valid email address', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmail@com",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

iv) Tests for Valid Passwords.

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmail@com",
            "password" : "A1_",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmail@com",
            "password" : "nfm_1",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmail@com",
            "password" : "nFm_@",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for Enter Valid Password', async () => {
    const req = {
        body: {
            "username":"Vyom123",
            "email" : "abc@gmail@com",
            "password" : "nFm_@",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

v)Test for checking existing tracks in a particular year.

```
test('should return status 300 for Track already exist in same year', async () => {
    const req = {
        body: {
            "username":"vyom12",
            "email" : "abc@gmail.com",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "trackdev123",
                    "start_date": "2022-04-22T20:09:27.848+00:00",
                    "end_date": "2022-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(300);
});
```

vi)Test for checking empty fields while signing.

```
test('should return status 500 for Internal Server Error', async () => {
    const req = {
        body: {
            "username":"",
            "email" : "abc@gmail.com",
            "password" : "A1_a",
            "track_list":[
                {
                    "track_name": "new_track",
                    "start_date": "2023-04-22T20:09:27.848+00:00",
                    "end_date": "2023-04-22T20:09:27.848+00:00"
                }
            ],
            "resume_link": "abc"
        }
    };
    const res = await request(baseURL).post('/api/organizer_signup').send(req.body);
    expect(res.status).toEqual(500);
});
```

vii)Tests for checking non-verified organizer login.

```
describe('POST /api/organizer_login', () => {
    test('should return status 400 for not verified organizer login', async () => {
        const req = {
            body: {
                username: 'Vyomxyz',
                password: 'A1_a'
            }
        };
        const res = await request(baseURL).post('/api/organizer_login').send(req.body);
        expect(res.status).toEqual(400);
    });
```

```
test('should return status 400 for Organizer not found', async () => {
    const req = {
        body: {
            username: 'vandan12',
            password: 'A1_a'
        }
    };
    const res = await request(baseURL).post('/api/organizer_login').send(req.body);
    expect(res.status).toEqual(400);
});
```

viii)Test for Invalid password.

```
test('should return status 400 for invalid password', async () => {
    const req = {
        body: {
            username: 'Vyom',
            password: 'abcd'
        }
    };
    const res = await request(baseURL).post('/api/organizer_login').send(req.body);
    expect(res.status).toEqual(400);
});
```

ix)Test for checking verified organizer login.

```
test('should return status 200 for verified organizer login', async () => {
    const req = {
        body: {
            username: 'Vyom12',
            password: 'A1_a'
        }
    };
    const res = await request(baseURL).post('/api/organizer_login').send(req.body);
    expect(res.status).toEqual(200);
});
```

Results of the above test cases:

```
  POST /api/organizer_signup
    √ should return status 200 for Successful signup (278 ms)
    √ should return status 300 for Username already exists (41 ms)
    √ should return status 400 for enter valid email address (35 ms)
    √ should return status 400 for enter valid email address (27 ms)
    √ should return status 400 for enter valid email address (44 ms)
    √ should return status 400 for enter valid email address (43 ms)
    √ should return status 400 for Enter Valid Password (38 ms)
    √ should return status 400 for Enter Valid Password (35 ms)
    √ should return status 400 for Enter Valid Password (26 ms)
    √ should return status 400 for Enter Valid Password (26 ms)
    √ should return status 300 for Track already exist in same year (56 ms)
    √ should return status 500 for Internal Server Error (91 ms)
  POST /api/organizer_login
    √ should return status 400 for not verified organizer login (388 ms)
    √ should return status 400 for Organizer not found (62 ms)
    √ should return status 400 for invalid password (39 ms)
    √ should return status 200 for verified organizer login (117 ms)

Test Suites: 1 passed, 1 total
Tests:       16 passed, 16 total
Snapshots:   0 total
Time:        2.367 s, estimated 3 s
Ran all test suites matching /organizer.test.js/i.
PS C:\Users\Vyom Patel\Desktop\proj\backend\test>
```

Login and SignUp for Teams:

i)Test for checking unique team entries.

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/team_signup', () => {
    test('should return status 300 for Team already exist for same name,year and track', async () => {
        const req = {
            body: {
                "team_name": "Team_Test_1",
                "track_name": "track_test",
                "track_year": "2023",
                "team_password": "Abc1_",
                "teammate_1": "Charmil"
            }
        };
        const res = await request(baseURL).post('/api/team_signup').send(req.body);
        expect(res.status).toEqual(300);
    });
```

ii)Test for checking if a track exists in the database.

```
test('should return status 300 for track dont exist in track db', async () => {
    const req = {
        body: {
            "team_name": "Team_Test_1",
            "track_name": "track_",
            "track_year": "2022",
            "team_password": "Abc1_",
            "teammate_1": "Charmil"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(300);
});
```

iii)Test for checking whether teammate-1 is a registered user.

```
test('should return status 500 for user not found of teammate-1', async () => {
    const req = {
        body: {
            "team_name": "Team_Test_2",
            "track_name": "track_test",
            "track_year": "2023",
            "team_password": "Abc1_",
            "teammate_1": "xyz"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(500);
});
```

iv)Tests for checking Valid Passwords.

```
test('should return status 400 for enter valid password', async () => {
    const req = {
        body: {
            "team_name": "Team_Test_2",
            "track_name": "track_test",
            "track_year": "2023",
            "team_password": "abc",
            "teammate_1": "Charmil"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
 test('should return status 400 for enter valid password', async () => {
     const req = {
         body: {
             "team_name": "Team_Test_2",
             "track_name": "track_test",
             "track_year": "2023",
             "team_password": "abc1_",
             "teammate_1": "Charmil"
         }
     };
     const res = await request(baseURL).post('/api/team_signup').send(req.body);
     expect(res.status).toEqual(400);
 });
```

```
test('should return status 400 for enter valid password', async () => {
    const req = {
        body: {
            "team_name": "Team_Test_2",
            "track_name": "track_test",
            "track_year": "2023",
            "team_password": "Abc1",
            "teammate_1": "Charmil"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

```
test('should return status 400 for enter valid password', async () => {
    const req = {
        body: {
            "team_name": "Team_Test_2",
            "track_name": "track_test",
            "track_year": "2023",
            "team_password": "Ab_",
            "teammate_1": "Charmil"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(400);
});
```

v)Test for checking registration is not possible.

```
test('should return status 500 for registration not possible', async () => {
    const req = {
        body: {
            "team_name": "Team_Test_2",
            "track_name": "track_test",
            "track_year": "2023",
            "team_password": "Ab_1",
            "teammate_1": "Charmil"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(500);
});
```

vi) Test for checking registration possible.

```
test('should return status 200 for Registration possible', async () => {
    const req = {
        body: {
                "team_name": "Team_Test_new",
                "track_name": "track_test",
                "track_year": "2023",
                "team_password": "Ab_1",
                "teammate_1": "meet123"
        }
    };
    const res = await request(baseURL).post('/api/team_signup').send(req.body);
    expect(res.status).toEqual(200);
});
```

vii)Test for checking verified user login.

```
describe('POST /api/team_login', () => {
    test('should return status 200 for verified login', async () => {
        const req = {
            body: {
                "team_name": "Team_Test_3",
                "team_password": "Ab_1"
            }
        };
        const res = await request(baseURL).post('/api/team_login').send(req.body);
        expect(res.status).toEqual(200);
    });
```

viii) Test for checking if a team is not found.

```
  test('should return status 400 for Team not found', async () => {
      const req = {
          body: {
              "team_name": "Team_Test3",
              "team_password": "Ab_1"
          }
      };
      const res = await request(baseURL).post('/api/team_login').send(req.body);
      expect(res.status).toEqual(400);
  });
```

ix)Test for checking Invalid Passwords.

```
test('should return status 400 for invalid password', async () => {
    const req = {
        body: {
            "team_name": "Team_Test3",
            "team_password": "Ab_"
        }
    };
    const res = await request(baseURL).post('/api/team_login').send(req.body);
    expect(res.status).toEqual(400);
});
```

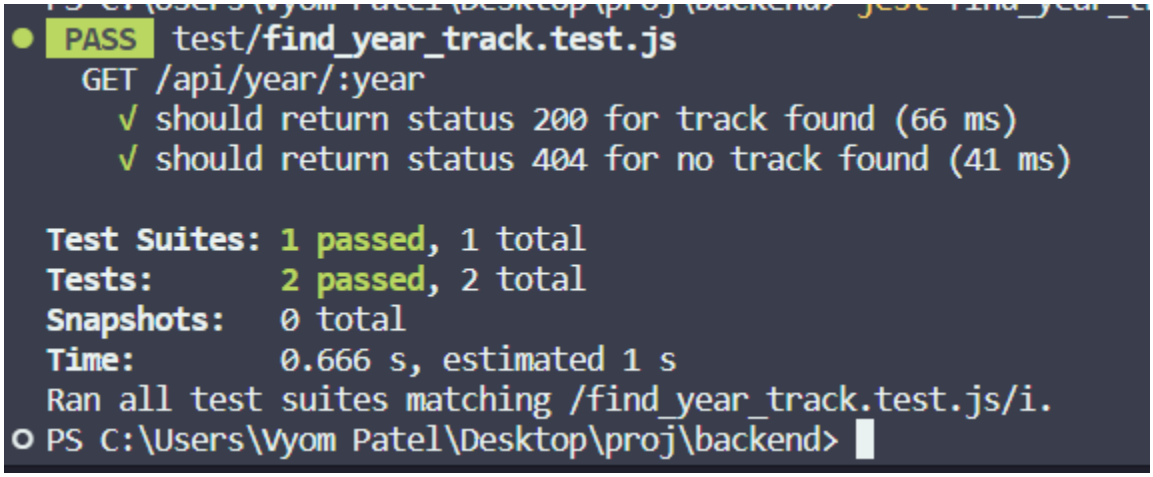
Results of the above test cases:

```
● PASS ./team.test.js
  POST /api/team_signup
    √ should return status 300 for Team already exist for same name,year and track (103 ms)
    √ should return status 300 for track dont exist in track db (65 ms)
    √ should return status 500 for user not found of teammate-1 (112 ms)
    √ should return status 400 for enter valid password (93 ms)
    √ should return status 400 for enter valid password (83 ms)
    √ should return status 400 for enter valid password (85 ms)
    √ should return status 400 for enter valid password (77 ms)
    √ should return status 500 for registration not possible (101 ms)
    √ should return status 200 for Registration possible (473 ms)
  POST /api/team_login
    √ should return status 200 for verified login (153 ms)
    √ should return status 400 for Team not found (31 ms)
    √ should return status 400 for invalid password (37 ms)

Test Suites: 1 passed, 1 total
Tests:       12 passed, 12 total
Snapshots:   0 total
Time:        2.285 s, estimated 3 s
Ran all test suites matching /team.test.js/i.
○ PS C:\Users\Vyom Patel\Desktop\proj\backend\test>
```

Find track functionality testing.

```
//find year track
describe('GET /api/year/:year', () => {
    test('should return status 200 for track found', async () => {
        const res = await request(baseURL).get('/api/year/2021')
        expect(res.status).toEqual(200);
    });

    test('should return status 404 for no track found', async () => {
        const res = await request(baseURL).get('/api/year/2024')
        expect(res.status).toEqual(404);
    });
}
);
```

Result of the above test case.



i)Test for find year track functionality

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

//testing for find_track
describe('GET /api/track', () => {
    test('should return status 200 for track found', async () => {
        const res = await request(baseURL).get('/api/track/?year=2020&name_code=track_201')
        expect(res.status).toEqual(200);
    });

    test('should return status 404 for no track found', async () => {
        const res = await r any t(baseURL).get('/api/track/?year=2021&name_code=track_201')
        expect(res.status).toEqual(404);
    });
}
);
```

Result of the above test case:

```
● PASS ./find_track.test.js
    GET /api/track
      √ should return status 200 for track found (92 ms)
      √ should return status 404 for no track found (59 ms)

  Test Suites: 1 passed, 1 total
  Tests:       2 passed, 2 total
  Snapshots:   0 total
  Time:        0.787 s, estimated 1 s
  Ran all test suites matching /find_track.test.js/i.
○ PS C:\Users\Vyom Patel\Desktop\proj\backend\test>
```

Add_home functionality.

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/add_home', () => {
    test('should return status 200 for successfull registration', async () => {
        const req = {
            body: {
                "year": "2017",
                "img": "img_2018",
                "WelcomeContent": {
                    "title": "Welcome_content_title_2021",
                    "content": " Welcome_content_content_2021"
                },
                "content": {
                    "keyNoteSpeakers": {
                        "title": "content_keynotespeakers_title_2021",
                        "list": [
                            {
                                "text":
"content_keynotespeakers_list_text_2021",
                                "link":
"content_keynotespeakers_list_link_2021"
                            },
                            {
                                "text":
"content_keynotespeakers_list_text_2021_2",
                                "link":
"content_keynotespeakers_list_link_2021_2"
                            }
                        ]
                    },
                    "invitedSpeakers": {
                        "title": "content_invitedspeakers_title_2021",
                        "list": [
                            {
                                "text":
"content_invitedspeakers_list_text_2021",
                                "link":
"content_invitedspeakers_list_link_2021"
                            },
                            {
                                "text":
"content_invitedspeakers_list_text_2021_2",
```

```
                                "link":
"content_invitedspeakers_list_link_2021_2"
                            }
                        ]
                    },
                    "tracks": {
                        "title": "content_tracks_title_2021",
                        "list": [
                            {
                                "text": "trackdev123",
                                "link": "jaymataji"
                            }
                        ]
                    },
                    "tutorials": {
                        "title": "content_tutorials_title_2021",
                        "list": [
                            {
                                "text": "content_tutorials_list_text_2021",
                                "link": "content_tutorials_list_link_2021"
                            },
                            {
                                "text": "content_tutorials_list_text_2021_2",
                                "link": "content_tutorials_list_link_2021_2"
                            }
                        ]
                    }
                }
            }
        };
        const res = await request(baseURL).post('/api/add_home').send(req.body);
        expect(res.status).toEqual(200);
    });

    test('should return status 300 for unsuccessfull registration', async () =>
{
        const req = {
            body: {
                "year": "2020",
                "img": "img_2019",
                "WelcomeContent": {
                    "title": "Welcome_content_title_2021",
                    "content": " Welcome_content_content_2021"
                },
                "content": {
                    "keyNoteSpeakers": {
                        "title": "content_keynotespeakers_title_2021",
```

```
                        "list": [
                            {
                                "text":
"content_keynotespeakers_list_text_2021",
                                "link":
"content_keynotespeakers_list_link_2021"
                            },
                            {
                                "text":
"content_keynotespeakers_list_text_2021_2",
                                "link":
"content_keynotespeakers_list_link_2021_2"
                            }
                        ]
                    },
                    "invitedSpeakers": {
                        "title": "content_invitedspeakers_title_2021",
                        "list": [
                            {
                                "text":
"content_invitedspeakers_list_text_2021",
                                "link":
"content_invitedspeakers_list_link_2021"
                            },
                            {
                                "text":
"content_invitedspeakers_list_text_2021_2",
                                "link":
"content_invitedspeakers_list_link_2021_2"
                            }
                        ]
                    },
                    "tracks": {
                        "title": "content_tracks_title_2021",
                        "list": [
                            {
                                "text": "trackdev123",
                                "link": "jaymataji"
                            }
                        ]
                    },
                    "tutorials": {
                        "title": "content_tutorials_title_2021",
                        "list": [
                            {
                                "text": "content_tutorials_list_text_2021",
                                "link": "content_tutorials_list_link_2021"
```

```
                            },
                            {
                                "text": "content_tutorials_list_text_2021_2",
                                "link": "content_tutorials_list_link_2021_2"
                            }
                        ]
                    }
                }
            }
        };
        const res = await request(baseURL).post('/api/add_home').send(req.body);
        expect(res.status).toEqual(300);
    });
});
```

Result of the above test case.

```
 PASS  test/add_home.test.js
  POST /api/add_home
    √ should return status 200 for successfull registration (191 ms)
    √ should return status 300 for successfull registration (47 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        0.886 s
Ran all test suites matching /add_home.test.js/i.
PS C:\Users\Vyom Patel\Desktop\proj\backend>
```

i)Test for checking Update track:

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/update_track', () => {
    test('should return status 200 for Update Successful', async () => {
        const req = {
            body: {
                    "name_code": "trackdev123",
                    "year": "2022",
```

```
"sidebar": [
  {
    "title": "Home",
    "links": [
      {
        "name": "Home"
      },
      {
        "name": "Track-Details"
      }
    ]
  }
],
"importantDates": {
  "title": "Contest Duration",
  "dates": [
    {
      "date": "22/5/2023",
      "event": "Start  the contest"
    }
  ]
},
"content": {
  "introduction": {
    "title": "Introduction",
    "content": "This is content of Introduction."
  },
  "TaskDescription": {
    "title": "Task Description",
    "content": "This is content of Task Description"
  },
  "corpus": {
    "title": "Training Corpus",
    "content": "Content of Training Corpus"
  },
  "registration": {
    "title": "Registration",
    "content": "Content of Registration"
  },
  "submission": {
    "title": "Submission Format",
    "content": "Content of Submission."
  },
  "evaluation": {
    "title": "Evaluation ",
    "content": "Content of Evaluation"
  }
```

```
                    },
                    "tag": [
                      {
                        "tagname": "ML"
                      }
                    ]
                }
        };
        const res = await
request(baseURL).post('/api/update_track').send(req.body);
        expect(res.status).toEqual(200);
    });
})
```

Result of the above test case.

```
 PASS  ./update_track.test.js
   POST /api/update_track
     √ should return status 200 for Update Successful (98 ms)

  Test Suites: 1 passed, 1 total
  Tests:       1 passed, 1 total
  Snapshots:   0 total
  Time:        0.727 s, estimated 1 s
  Ran all test suites matching /update_track.test.js/i.
PS C:\Users\Vyom Patel\Desktop\proj\backend\test>
```

ii)Test for checking Set_score functionality

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/set_score', () => {
    test('should return status 200 for Successful setting score', async () => {
        const req = {
            body: {
                "track_name":"trackdev123",
                "team_name":"meet123",
                "score": 1500,
                "track_year": "2022"
            }
        };
        const res = await request(baseURL).post('/api/set_score').send(req.body);
        expect(res.status).toEqual(200);
    })
})
```

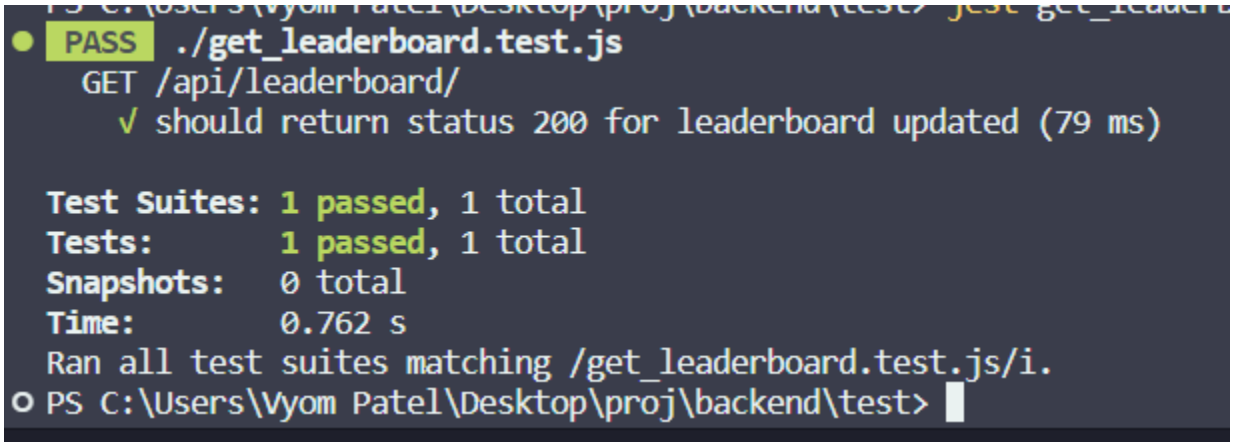
Result of the above test case.

```
   POST /api/set_score
     √ should return status 200 for Successful setting score (95 ms)

  Test Suites: 1 passed, 1 total
  Tests:       1 passed, 1 total
  Snapshots:   0 total
  Time:        0.73 s
  Ran all test suites matching /set_score.test.js/i.
PS C:\Users\Vyom Patel\Desktop\proj\backend\test>
```

iii)Test for checking Get_leaderboard functionality

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('GET /api/leaderboard/', () => {
    test('should return status 200 for leaderboard updated', async () => {
        const res = await request(baseURL).get('/api/leaderboard/?track_name=trackdev123&track_year=2022')
        expect(res.status).toEqual(200);
    });
})
```

Result of the above test case.



iv)Test for verifying if the track gets added to the database by admin.

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/add_track_admin', () => {
    test('should return status 200 for track added to track db added by admin',
async () => {
        const req = {
            body: {
                    "name_code": "track_vyom",
                    "year": "2022",
                    "sidebar": [
```

```
    {
      "title": "Home",
      "links": [
        {
          "name": "Home"
        }
      ]
    }
  ],
  "importantDates": {
    "title": "Contest Duration",
    "dates": [
      {
        "date": "22/5/2023",
        "event": "Start  the contest"
      }
    ]
  },
  "content": {
    "introduction": {
      "title": "Introduction",
      "content": "This is content of Introduction."
    },
    "TaskDescription": {
      "title": "Task Description",
      "content": "This is content of Task Description"
    },
    "corpus": {
      "title": "Training Corpus",
      "content": "Content of Training Corpus"
    },
    "registration": {
      "title": "Registration",
      "content": "Content of Registration"
    },
    "submission": {
      "title": "Submission Format",
      "content": "Content of Submission."
    },
    "evaluation": {
      "title": "Evaluation ",
      "content": "Content of Evaluation"
    }
  },
  "tag": [
    {
      "tagname": "ML"
```

```
                    }
                ]
            }
        };
        const res = await
request(baseURL).post('/api/add_track_admin').send(req.body);
        expect(res.status).toEqual(200);
    }),
    test('should return status 500 for same track name error', async () => {
        const req = {
            body: {
                "name_code": "track_vyom",
                "year": "2022",
                "sidebar": [
                  {
                    "title": "Home",
                    "links": [
                      {
                        "name": "Home"
                      }
                    ]
                  }
                ],
                "importantDates": {
                  "title": "Contest Duration",
                  "dates": [
                    {
                      "date": "22/5/2023",
                      "event": "Start  the contest"
                    }
                  ]
                },
                "content": {
                  "introduction": {
                    "title": "Introduction",
                    "content": "This is content of Introduction."
                  },
                  "TaskDescription": {
                    "title": "Task Description",
                    "content": "This is content of Task Description"
                  },
                  "corpus": {
                    "title": "Training Corpus",
                    "content": "Content of Training Corpus"
                  },
                  "registration": {
                    "title": "Registration",
```

```
                        "content": "Content of Registration"
                      },
                      "submission": {
                        "title": "Submission Format",
                        "content": "Content of Submission."
                      },
                      "evaluation": {
                        "title": "Evaluation ",
                        "content": "Content of Evaluation"
                      }
                    },
                    "tag": [
                      {
                        "tagname": "ML"
                      }
                    ]
                }
        };
        const res = await
request(baseURL).post('/api/add_track_admin').send(req.body);
        expect(res.status).toEqual(500);
    })
})
```

Result of the above test case.

```
 PASS  ./add_track_admin.test.js
  POST /api/add_track_admin
    √ should return status 200 for track added to track db added by admin (254 ms)
    √ should return status 500 for same track name error (77 ms)

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        0.981 s, estimated 1 s
Ran all test suites matching /add track admin.test.js/i.
```

v)Test for checking track name in track database.

```
const request = require('supertest');

const baseURL = "http://localhost:5000";

describe('POST /api/add_track_organizer', () => {
    test('should return status 300 for track name already exist in track dbs',
async () => {
```

```
        const req = {
            body: {
                "username" : "Vyomxy",
                "track_name" :"trackdev123",
                "start_date" : "2022-04-24T00:00:00.000+00:00",
                "end_date" : "2022-05-24T00:00:00.000+00:00"
            }
        };
        const res = await
request(baseURL).post('/api/add_track_organizer').send(req.body);
        expect(res.status).toEqual(300);
    }),
    test('should return status 200 for track added for verification
successfully', async () => {
        const req = {
            body: {
                "username" : "Vyomxy",
                "track_name" :"track_123",
                "start_date" : "2022-04-24T00:00:00.000+00:00",
                "end_date" : "2022-05-24T00:00:00.000+00:00"
            }
        };
        const res = await
request(baseURL).post('/api/add_track_organizer').send(req.body);
        expect(res.status).toEqual(200);
    }),
    test('should return status 400 for You already requested for this track in
this year, please wait for admins approval.', async () => {
        const req = {
            body: {
                "username" : "Vyomxy",
                "track_name" :"track_123",
                "start_date" : "2022-04-24T00:00:00.000+00:00",
                "end_date" : "2022-05-24T00:00:00.000+00:00"
            }
        };
        const res = await
request(baseURL).post('/api/add_track_organizer').send(req.body);
        expect(res.status).toEqual(400);
    })
})
```

Result of the above test case.