uCredit Backend Testing

Brayden Archer, barcher9
Matthew Liu, mliu78

In this project, we were able to increase code testing coverage of almost all test files we implemented to over 90% for statement, branch, function, and line coverage from a practically untested backend codebase over the course of 167 tests.

1. Testing Tools and Artifacts

The testing tool we used in this project is the Jest Testing library and the Jest supertest for both unit tests route integration tests.

2. SUT

The express backend repository of uCredit, the degree planning app we are testing. We chose the backend of this app, as one of the group members have worked a bit on the frontend in the past. The repository below is a forked version of the official backend repository that we created tests on.
https://github.com/mliu78/uCredit-API

3. Test Files, Configuration Files, Tools

Test files include all .spec.js files in the tests folder:

- course.spec.js
- distribution.spec.js
- evaluation.spec.js
- experiments.spec.js
- majors.spec.js
- plan.spec.js
- planReview.spec.js
- search.spec.js
- sisData.spec.js
- SSO.spec.js
- user.spec.js
- year.spec.js
- distributionCreditUpdate.js
- distributionSamples.js
- experimentDAO.spec.js
- userSamples.spec.js

Configuration files include:

- package.json
- jest.config.js
- jest-mongodb-config.js
- .env

All files except the .env files should be included in the repository. You will have to create the .env file yourself in the root directory. Instructions below.

As mentioned before, the testing tool we used is the jest testing library

4. Presentation Slides

https://docs.google.com/presentation/d/1RORz4lgao8wrJCyd7ncSTC-8jTmMermjQ507cG31-lU/edit?usp=sharing

5. Instructions on Running Tests in Demo

1. Clone github repository
2. Make sure Mongodb, Node.js, and npm are installed on your computer
2. Add the secrets below to a .env file
2. Run “npm i” in folder of repository
3. Run “npm test”
4. Tests will run and report in terminal

Secrets (copy below to your .env file in your root directory):
DB_ADMIN_PASSWORD = "y0XQwI7fISy8U7kX" SIS_API_KEY = "c6DTXIBAujqzphi1T19D8pn3qajbp8BX" DOMAIN = "http://localhost:4567" SESSION_SECRET = "T$9s5~s%Qh`Y}n{'" PUBLIC_KEY = "-----BEGIN CERTIFICATE-----
MIIFRDCCAywCCQDpZIys9tTaODANBgkqhkiG9w0BAQsFADBkMQswCQYDVQQGEwJ1
czERMA8GA1UECAwIbWFyeWxhbmQxEjAQBgNVBAcMCWJhbHRpbW9yZTEQMA4GA1UE
CgwHdWNyZWRpdDEcMBoGCSqGSIb3DQEJARYNcXd1MjlAamh1LmVkdTAeFw0yMTAy
MjQwODMzMDdaFw0yMjAyMjQwODMzMDdaMGQxCzAJBgNVBAYTAnVzMREwDwYDVQQI
DAhtYXJ5bGFuZDESMBAGA1UEBwwJYmFsdGltb3JlMRAwDgYDVQQKDAd1Y3JlZGl0
MRwwGgYJKoZIhvcNAQkBFg1xd3UyOUBqaHUuZWR1MIICIjANBgkqhkiG9w0BAQEF
AAOCAg8AMIICCgKCAgEArGwDB1xFUA4R+EX58Gqd+p4ZNI/CsOHEKTBevv0r36Yd
FcQv1QpKE/MRZ5bUFfSqXI5rTNmCvGObrHZ8zIX9WG1tSdJz/3MuNxxICoVOD2rG
L0fs+I1aXDy3BNgAV7+Z/Jm0Fryt8rkXgkbH9Kiw1lvuuk2cre2lxMy0M/BIsBSg
qh9BtjXkwbl8s6WWsBBbAe/nzXZ1QrDMoV8883Q/kGi7XKmDxCRwQvJKaTlNWe9f
rY8GL9vWIs5TrAHLnB+BbF37FJ0YolGuV0BueXH0iVCV6g+KdjEFDCjMx9zSDn1d
9M98Nblrn5rx7Z2lCVlUZWiBNBd7xmy6Qw6AhwNwHIIXMqYsiVt1hyatbgRqJKdH
nSxcpi7O1eVR9Z4TfhmG9J10nWqzLL2ZCinTwwZ1kL+iu9lm2adg+zi9Wj8mwmAU
lgSigmvn0fohUKEPEeTnRCSpVPKU3Co9uwu8Z09PIaO43U4k8mZA+4cm9HSK9ABM

YASehK1+r2om622cOZbPVF3nVlw6EzoBk8Z9ZFRIRFRW/jsE6RSGlH0KZ7fn2mqw
K4ITcyo3eucEBK2ju+WzpW32HhsPMzBZ45a/zScxh5BB2itTR3tQ4DDKTtvpcmJJ
sa2XZS+AaSZVVT96JBvAtQIsS7SKFoVJ0Nb+ivDTASEa/Zi6Z3gXHsB2ghhWf2kC
AwEAATANBgkqhkiG9w0BAQsFAAOCAgEAV3ViU0qhPDv2XNEmDJcxrFf6x5WwPeL6
AFur6XGMO2Wuvs+/Kio2xRbJma2OI7YRirzKo660kTB2XgneN0kb93B67UFdK7Gm
HT02S1OotZCpxA3jfFANL5GzK62S8QSCmtXAEiW/lzU/aocj2hpJrHIJbzsoNRM/
qgdV68QiIIKgbbI7ryE3y8gDne1KEIK+tZc3rVBI9ALpksgZo7SxeIlXy2oNVirR
x5u9puJEzwoqkQWuPKsp9cFDR7t42RIcC55ivpt/aMdaVrLWSKzw9RqJIJATE2l5
AbiF7TXVdsDTfxyloIvRW5e4uqkhsoPNkmrjIlKQhowuLp3p6xxyN3vGrAEkDVSk
7vl9W6VQYMBsqHc+xIy7MUa3FW0KQcut4vAskaNZ4dWsN6LrqOEr47/Ni+3abloQ
gY29BtBsYwPt9eOfIM2002s94pkgVSPgr7h5hVmIRS+vgVQlwIXCgEWTEiuz+GDy
3kdbNFHkLkYsjX7l4/QPM6uFF+CZnYWqY/lemWXZgPiM2L1GjYKW5Vl6EjbYb1so
uy8ZEnHnt+Pk0IyrsdYxEGlvOmJEg/0mvDDyP0u2yXj0G0vKb4intI4NxX5LTwIX
DwrZ3wXa4XICGGClQ/g+TuIGOQDwVRl8cenAfik1/g264dQgiX+CN0mUAom8qOdV
gwwXpjgNk6k= -----END CERTIFICATE-----" PRIVATE_KEY = "-----BEGIN PRIVATE KEY-----
MIIJQwIBADANBgkqhkiG9w0BAQEFAASCCS0wggkpAgEAAoICAQCsbAMHXEVQDhH4
Rfnwap36nhk0j8Kw4cQpMF6+/Svfph0VxC/VCkoT8xFnltQV9KpcjmtM2YK8Y5us
dnzMhf1YbW1J0nP/cy43HEgKhU4PasYvR+z4jVpcPLcE2ABXv5n8mbQWvK3yuReC
Rsf0qLDWW+66TZyt7aXEzLQz8GWwFKCqH0G2NeTBuXyzpZawEFsB7+fNdnVCsMyh
XzzzdD+QaLtcqYPEJHBC8kppMg1Z71+tjwYv29YizlOsAcucH4FsXfsUnRiiUa5X
QG55cfSJUJXqD4p2MQUMKMzH3NIOfV30z3w1uWufmvHtnaUJWVRlaIE0F3vGbLpD
DoCHA3AciVcypiyJW3WHJq1uBGokp0edLFymLs7V5VH1nhN+GYb0nXSdarMsvZkK
KdPDBnWQv6K72WbZp2D7OL1aPybCYBSWBKKCa+fR+iFQoQ8R5OdEJKlU8pTcKj27
C7xnT08ho7jdTiTyZkD7hyb0dIr0AExgBJ6ErX6vaibrbZw5ls9UXedWXDoTOgGT
xn1kVEhEVFb+OwTpFIaUfQpnt+faarArghNzKjd65wQEraO75bOlbfYeGw8zMFnj
lr/NJzGHkEHaK1NHe1DgMMpO2+lyYkmxrZdlL4BpJlVVP3okG8C1AixLtIoWhUnQ
1v6K8NMBIRr9mLpneBcewHaCGFZ/aQIDAQABAoICAHKBTMWflnxiy7ZIqoLqGz4r
rmuEqXQUitQbxmAp+/AL/jbNUkK2EZoMC1vzA6gNEvJitomTzcnUkLbI/XpJ5YTL
LxAejmBwGpoI0yMBgmSksTcHGi26me61d4nk9N1RZi8l2D5dUVvnZeBjpzkoqj/B
T9oz3sB3GWaH14jCtpVoLYatOVOLmP66c0FOz7LfOcEglJ2TdK40gcENYYoAgpIh
q4QPI22oEVG0XIxd7BX07WJyoFY6NYxp35FUo+Zr2eSgyTTYW1q4ygIxzOwkLJTg
ip6lcXuqdEKt+cklMe7jfPkurbt0BzA4k0ANOGPsCitKaruweMe+vzf3Mo33T4Tc
9zGdXx3JaBRIobG14/BkRkSEA6kHc3lJF3dYIrfj12gKFmYomUVmVLiaB2lhEvGp
6ggj4ISh14so8Squpd3+zNS9VCE9ovo3VB5uUoT+tRd0SAs3Ds7s4Pik22JGCxj6
snN9aty23iLnMUDg2qtYITkIdxrDj7b3hpzOeCzW66iMdD36yUStcFYsF2bLMCyz
QU7LMO7BXT4OjRa3M5SnYwV+mQ7UUxzEvRcIL1NRcLhgN8PhrFtis83ZXRHHdlRn
z8riQ0WEAPQIoEUrc11A0NqXGKhbz596usBp7EHOijizPa5OAuhpTwn9vn2j65vr
Uz/RaSNzBBJIF6X9NdD9AoIBAQDShrnyQIo0kAn8Zl26urTrvzu9uGtq5AifECtb
fA36nxnx/JcBEMpcyVEgndzJsSv6OA40/7/ozSLUW0U2jV116qXNDQt+SOJrVxb5
SPI0m7OkHDMN9PMhSNJCwhBG+ucNMVb5UTpcTYhtVGpgZ4eLWwWWMCdte+iS7iBm
Vw/mGjjNsDaBIErIdJ7SndxF9vb1xhaDu/efKfhKmliLQgtPliZaIoPl4ipon9pj
z92Ky3/QAx2TK3wngQJlqEmol/BRKjKcLc3k6ouwhGDyUVJniB0roGmsD7KuDILc
E721lgOKrLjXVJV3rc+RffeRBlMq915Ke4oUdiJpnKfQv2fHAoIBAQDRqkPMGSbG
igbGZvogiYmTmim3e4a2XLsYjMJB05et7etOkpqnxkmnHhyl8lxF//qz7POLNl2a

M9iT1jYvcjv6cjHk3eW71DEZODgTj5cjTewTdLUnJLmKRBnM46QJVqnyPPnP+ayh
JbZS3tmREVXjkrAvpXoMzkDy0ooS1Y0TtB3i1BlrvIEb6MzDv+SlZUWn1+HfS6k1
b8/uEi0Kjd7OEtVF+JuqIGfgFdqyuWKVKVHrkQslKNzg7+L3laOwbmH/EPTvlHr5
WVYuDbL6HQ2D2JmDdRO1ge1yRA//aiqBISO4jOkl8CLFKKb/I7oUtYj25BuxdOWu
i02XbeaaaL9PAoIBAQC819NGRbRzT91VRfG/xSjy6zhIh3v7hw9gt11dE9tqO3tH
gL9nWktbjc4zBsf9N+rt6Yh87AiWaCRc/n4IYCg81L3jQFdV8UUA6j8WX+O7Ywty
I0V8uIO7EJpi/bciJIBLaj3NKJrEH1xITdeaIGQKkhxhy1IiuGo4Glh/1IrTaiQY
BX/utiu+wjWRUb88FzsR3Z1mj56gbEJnkSnpyNVij5pZceqEtCo8bV+19siQbth4
m8LUt8YHnYar75gbHieYdtStGb0+IVIqhaR82IuliQg5kQwZJn+t9lUP2rlqgGUu
6J5psVrd3tB5ftgqg9TojrpK76Q9lHZr6/TwoDETAoIBAQCABmnTrOACh7HPYH4H
dDRvzsj+Yw1FOaZ9PSY5L80ExazTk62gGPJMe5SeuDkj/UTGrT+hmBbb3CC5VSpN
GoIJUcWprN1lLhK1BaEoQJvXUPOAhdlAB6rMBP2EINS/Zw3q/tTD6/8/f1mvjKhp
bp51kdLHpVG7lA4QuALbsU6t42QEZ+MF6FsmAadXOEuTux0neilQQlEWcioTg0HB
mhOc3d8hFMdowNTs/itGyvCpiEuffjfB/wuwxopfCB6l0yE/sMj/qjjVHXoEMRVC
qI7iHVbP8KOSR74H8guauOcvZl77h/+tzM5OeZ5PWS99CcSSjYNjfq/pPuKoPtF7
l879AoIBAA5m1tCOJ4bjHMf0y1dgGzfcD9OrxLAqcDm/2WP0ElZoTSsGVT6V3wY/
BfpVsHsCx3Moij1PuH18ZPuEot8ZEv9J0tGAHxgDHEzVPhrgGhaiH1GKj8mrewih
3z+yguVy6gf86FZwURdhVdYAJH08RIwdPR0Osq3q8dB+78pR/a4/qLVBXgLZQWza
T4wQxb4hWhbska0TyJdOrREOGUJgGkpbpj+48QujpkZnQbZ0nSwNdqXmprPQq2TY
yr1sB33oHLfb8bqfLXncakjN8qGLvhQQ/kld3/4yl6RqOpIzw5ccWoRD2hjreguD
5A6N7fputNzEp1DNzhis+gH160P+3yM= -----END PRIVATE KEY-----"

6. Screenshots of Running Tests

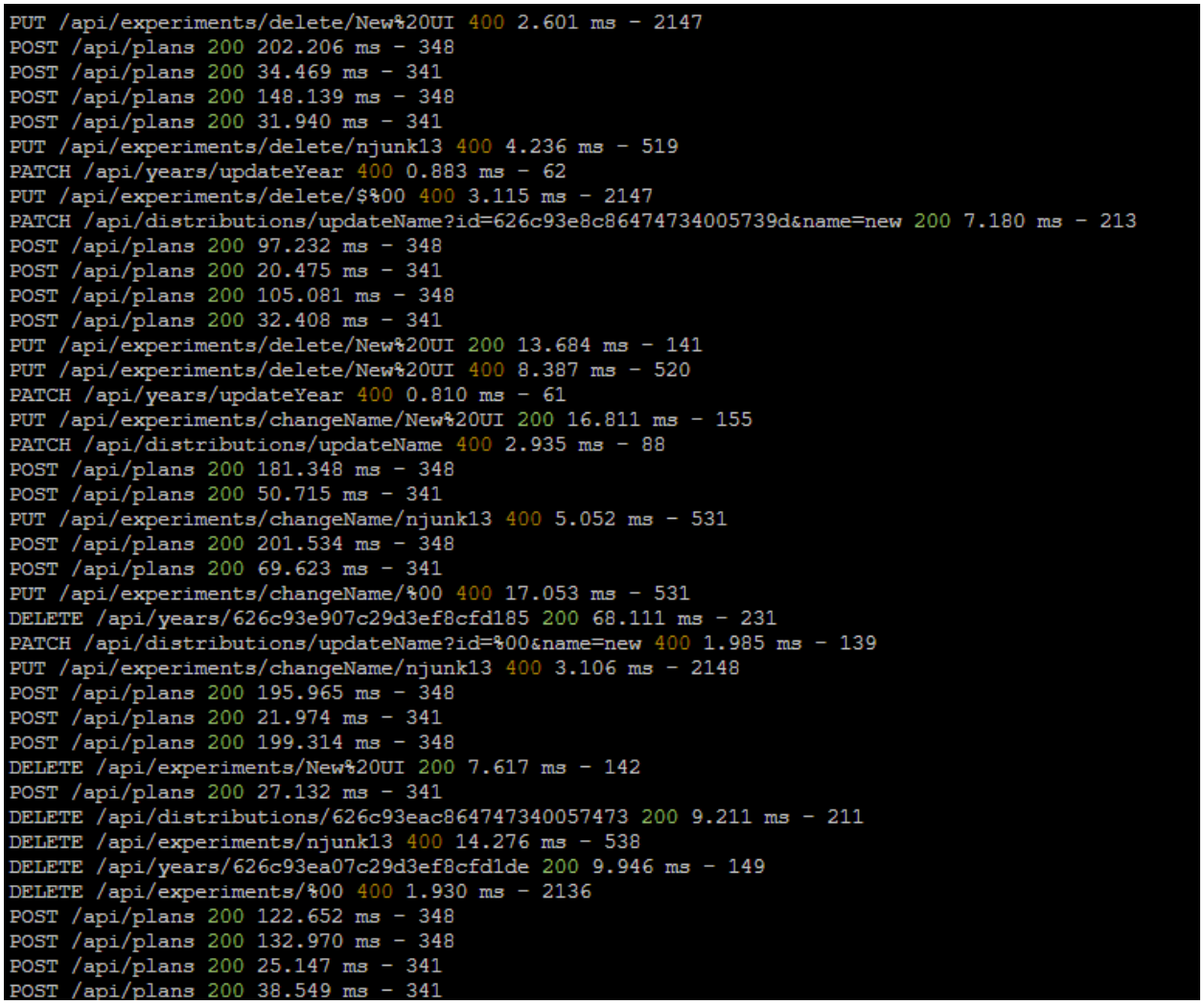


Figure 1: screenshot of backend route testing making requests to specific routes.

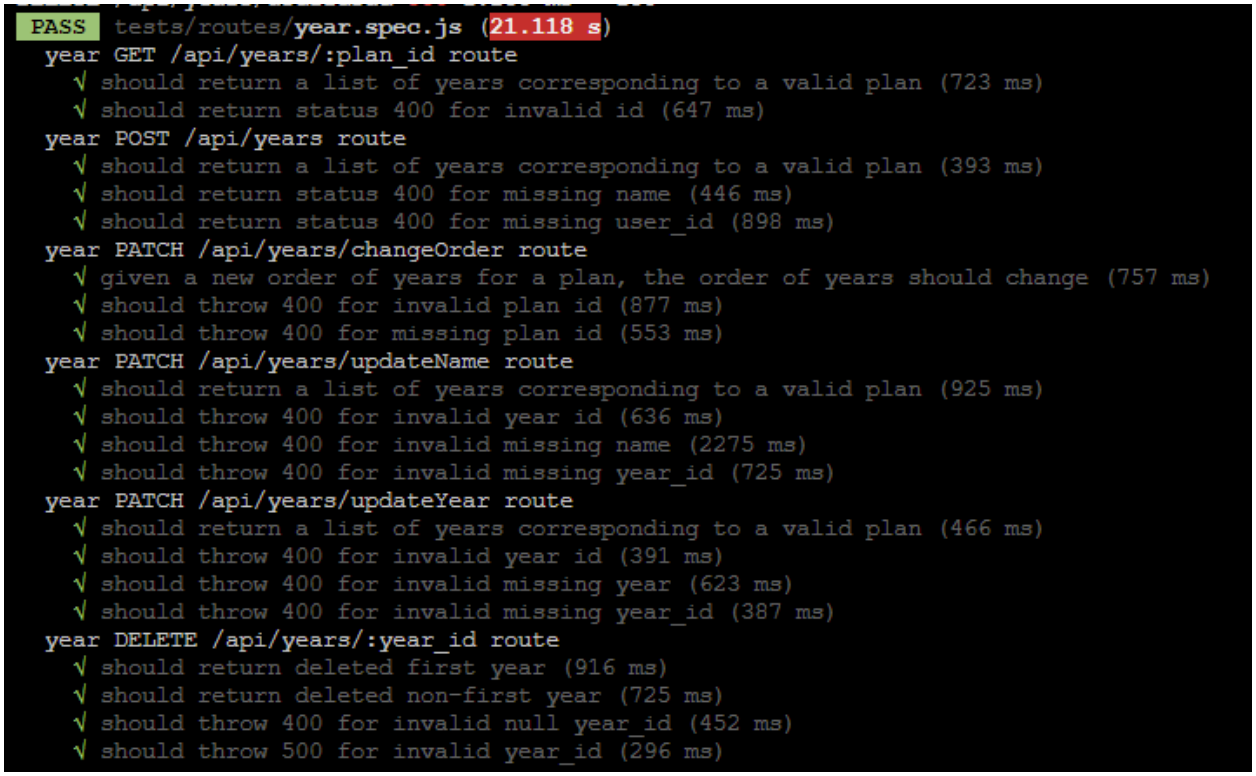


Figure 2: Image of passing and failing tests for a particular test file

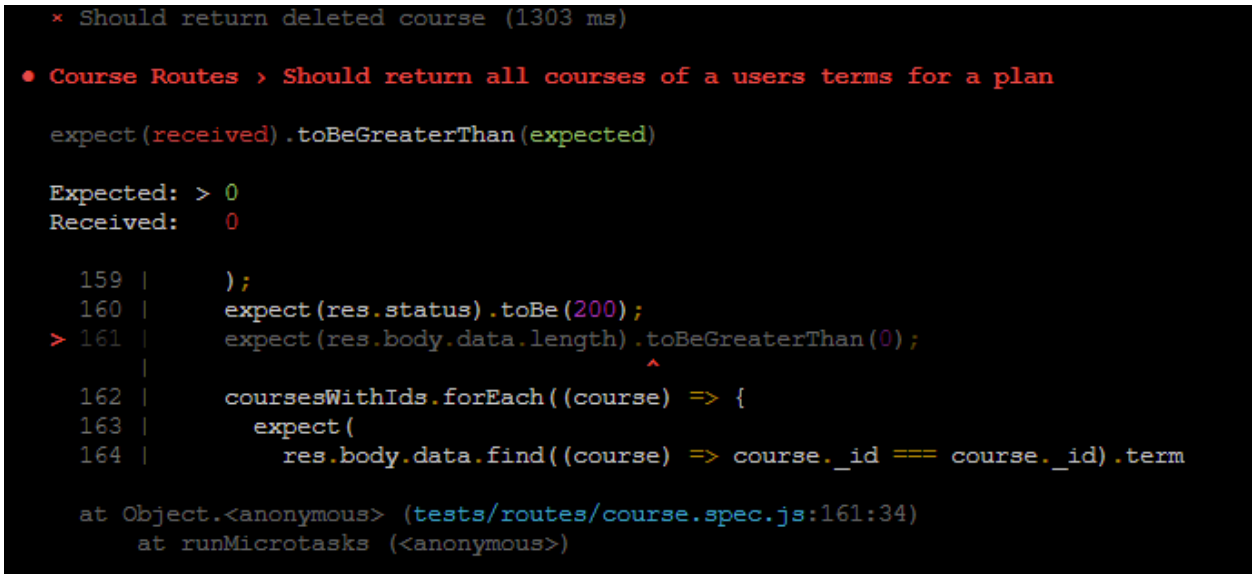
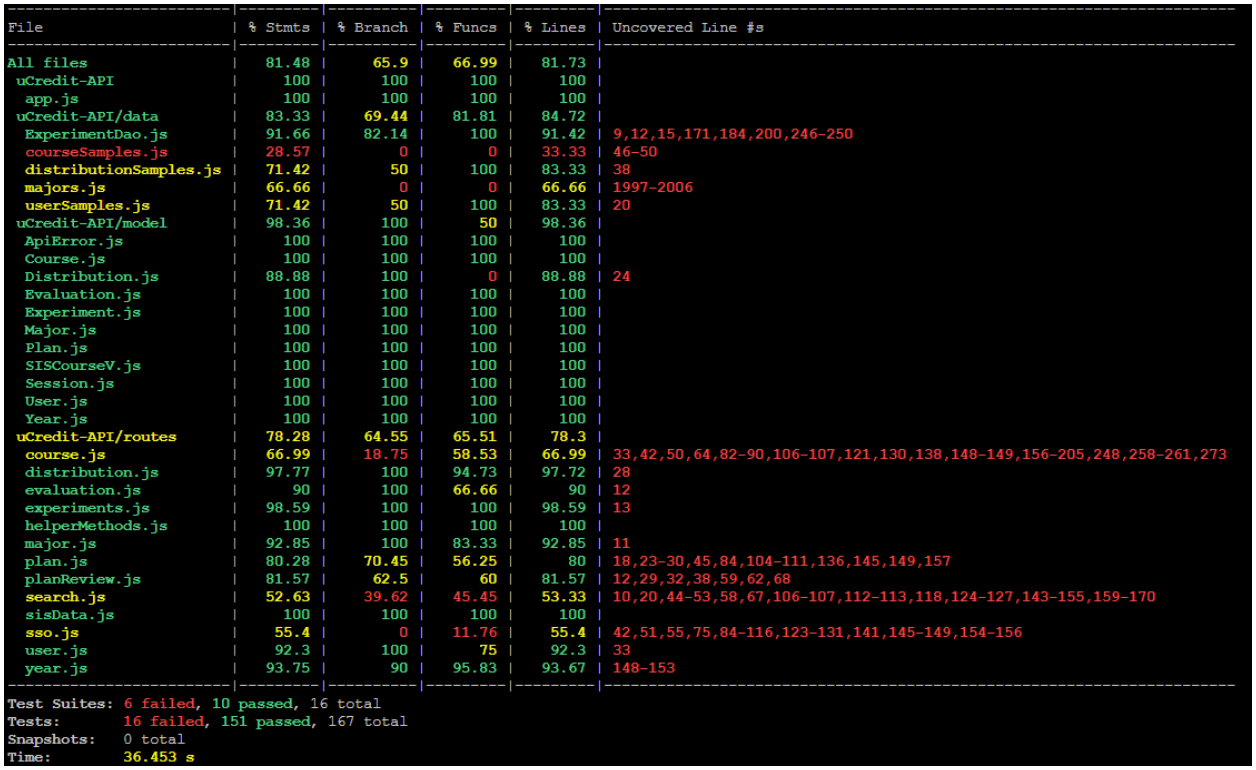


Figure 3: Image of failed test.



File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	81.48	65.9	66.99	81.73	
uCredit-API	100	100	100	100	
app.js	100	100	100	100	
uCredit-API/data	83.33	69.44	81.81	84.72	
ExperimentDao.js	91.66	82.14	100	91.42	9,12,15,171,184,200,246-250
courseSamples.js	28.57	0	0	33.33	46-50
distributionSamples.js	71.42	50	100	83.33	38
majors.js	66.66	0	0	66.66	1997-2006
userSamples.js	71.42	50	100	83.33	20
uCredit-API/model	98.36	100	50	98.36	
ApiError.js	100	100	100	100	
Course.js	100	100	100	100	
Distribution.js	88.88	100	0	88.88	24
Evaluation.js	100	100	100	100	
Experiment.js	100	100	100	100	
Major.js	100	100	100	100	
Plan.js	100	100	100	100	
SISCourseV.js	100	100	100	100	
Session.js	100	100	100	100	
User.js	100	100	100	100	
Year.js	100	100	100	100	
uCredit-API/routes	78.28	64.55	65.51	78.3	
course.js	66.99	18.75	58.53	66.99	33,42,50,64,82-90,106-107,121,130,138,148-149,156-205,248,258-261,273
distribution.js	97.77	100	94.73	97.72	28
evaluation.js	90	100	66.66	90	12
experiments.js	98.59	100	100	98.59	13
helperMethods.js	100	100	100	100	
major.js	92.85	100	83.33	92.85	11
plan.js	80.28	70.45	56.25	80	18,23-30,45,84,104-111,136,145,149,157
planReview.js	81.57	62.5	60	81.57	12,29,32,38,59,62,68
search.js	52.63	39.62	45.45	53.33	10,20,44-53,58,67,106-107,112-113,118,124-127,143-155,159-170
sisData.js	100	100	100	100	
sso.js	55.4	0	11.76	55.4	42,51,55,75,84-116,123-131,141,145-149,154-156
user.js	92.3	100	75	92.3	33
year.js	93.75	90	95.83	93.67	148-153

Test Suites: 6 failed, 10 passed, 16 total
Tests: 16 failed, 151 passed, 167 total
Snapshots: 0 total
Time: 36.453 s

Figure 4: Image of overall coverage information.

7. Unit Testing Information

The following unit tests below were performed via blackbox testing

1. ExperimentDAO.spec.js

Base case testing was used to verify valid and invalid inputs to check error edge cases. Testing was performed by documentation provided via chat messages with the developer of the app.

The following unit tests below were performed via whitebox testing:

1. distributionCreditUpdate.spec.js
2. distributionSamples.spec.js
3. userSamples.spec.js

Base case testing was used to verify valid and invalid inputs to check error edge cases.

We were initially going to perform unit testing on all files in the data file. However, we noticed that the majority of the files required connection to the production database. As we didn't want to possible perform any modifications on the production database, we erred on the side of caution and performed testing on the safe files, the files and functions associated with the tests above. Even so, we were able to increase overall coverage from less than 50% for statement, cranch, function, and lines to 83.33%, 69.44%, 81.81%, and 84.72% respectively, as seen in Figure 4 above.

8. API Testing Information

Although we weren't able to thoroughly test all helper functions and files, we were able to thoroughly test the backend api routes in over 130 tests. As seen in Figure 4 above, of the coverage information of the tests, we were able increase overall coverage of the routes from less than 10% for statement, cranch, function, and lines to 78.28%, 64.55%, 65.51%, and 78.3% respectively.

9. Main Faults Information

Error Handling: Specific cases of information could cause a 200 status code but error out after the status was returned, causing the header to be set after the response was returned to the client. The solution is to create more specific error cases instead of having a catch all method.

Note: Sometimes there are issues with asynchronous jest actions. Please run the test once or twice if the failing tests below don't match up.

Failing Tests below:

1. **search.spec.js:** Search routes
   Fail due to limited access to production database. There is little documentation on setting up the database with SIS courses and thus these tests could not be performed. Tests are passing when access to production database is accessible.
2. **course.spec.js:**
   a. **"Should return all courses of a users terms for a plan"**: CoursesByTerm fail in course.spec.jst: Empty array returned for valid query. Should return all courses of a users terms for a plan. Fails due to the improper packing of a search term variable during a method call. The solution to this fault is to wrap the matched term in curly braces on line 60 of routes/course.js.
   b. **"Should return deleted course":** Delete failed due to incorrect error handling for supertests.
3. **planReview.spec.js**:
   a. **"should create a new plan review and return the updated plan"**: user whitelisted_plan_ids is modified by attempting to push a plan id as a normal javascript array. However, arrays returned by MongoDB are sometimes of the MongoArray type and is thus unsafe to modify an array like this. Rather, findByIdAndUpdate with a $push into whitelisted_plan_ids would be the safer option
   b. "should delete a plan review and return the plan with the removed plan review": user whitelisted_plan_ids is modified by attempting to push a plan id as a normal javascript array. However, arrays returned by MongoDB are sometimes of the MongoArray type and is thus unsafe to modify an array like this. Rather, findByIdAndUpdate with a $push into whitelisted_plan_ids would be the safer option

4. **distributionSamples.spec.js:**
    a. **first distribution required credit count should match first sample required credit count**
    b. **second student required credit count should match for second sample required credit count**
    c. **third student required credit count should match for third sample required credit count**
    d. **fourth student required credit count should match for fourth sample required credit count**
    e. **fifth student required credit count should match for fifth sample required credit count**
    f. **sixth student required credit count should match for sixth sample required credit count**
    g. The tests above failed due to the fault where distributionSamples did not correctly add the “required” attribute to each sample distribution object.