

DRF PoC Web API

- <https://graphql-drf-poc-api-master.herokuapp.com/admingq/graphql/>
- Use above link to access GraphQL based server API.
- Username :- admin (lower case) – Admin
- Password :- poctest#1 (lower case)
- Username :- editor (lower case) – Non Admin
- Password :- test#123 (lower case)

Salient features

- Solution that allows insurers to define their own custom data model for their risks. There are no database tables called automobiles, houses, or prizes. Instead, insurers will be able to create their own risk types and attach as many different fields as they would like.
- Fields are bits of data like first name, age, zip code, model, serial number, Coverage A limit, or prize dollar amount. Basically any data the carrier would want to collect about the risk. Fields can also be of different types, like text, date, number, currency, and so forth.

Salient features (continued)

- As risk field types are user defined and not committed at time of table creation. Actual field/column value stored in table is string type. But field type sanctity is enforced using model validations. i.e User can not supply Date value where Currency is expected.
- Uniqueness among RiskType names is maintained by enforcing unique constrain. But enforcing unique risk field name in similar fashion is somewhat restrictive and naive. I have tried to maintain unique field name within Risk Types and not across all Risk Type using mode validations.

Salient features (continued)

- API allows users to create RiskTypes and associated RiskTypeFields in one go with use of nested serializer.
- During Risk creation system checks that proper referential integrity is maintained with Risk Type and during creation of Risk Fields system ensures that proper referential integrity is maintained with Risk and Risk Type fields.
- Risk API allows users to create Risk and associated RiskFields in one go with use of nested serializer.
- Various model validation errors in RiskType and RiskTypeFields, Risk and RiskFields are returned to client of serve API in JSON format.

Salient features (continued)

- Risk and RiskFields carry extra metadata related to RiskType and RiskTypeFields.
- On authentication front, only authenticated users can access server API. Role based authorization further enable only admin users to create/delete RiskType. Non admin users only have list and details option for RiskTypes. However they can create Risk instances based on RiskTypes.
- Session authentication is used for browsable API that runs within same session that of Web API
- Token authentication is used for separate website that access Web API for retrieving and creating RiskTypes and Risk.

Salient features (continued)

- Used Graphene Django library to expose all of API over single endpoint.
- Added new View class to the application so as to avoid unauthenticated access to GraphQL queries and mutation. Default GraphQL view in Graphene library allows access to all of the users.
- GraphQL has concept of queries i.e. Client application defines shape of data to be retried from server. Using Graphene-Django library we can define Query and Resolver classes. Thus root element in GraphQL query map to one of the resolver class which is further used to get and filter data from underlying Django model.
- GraphQL Mutations, Input types are used for adding and updating Django Models. Made use of Django serializers in DRF to do all validation before committing model changes.

Screenshot Heroku deployment

The screenshot shows the Heroku dashboard for the application 'graphql-drf-poc-api-master'. The browser address bar shows the URL 'dashboard.heroku.com/apps/graphql-drf-poc-api-master'. The Heroku logo is in the top left, and a search bar is in the top center. The app name is displayed in the top right, along with 'Open app' and 'More' buttons. The navigation menu includes 'Overview', 'Resources', 'Deploy', 'Metrics', 'Activity', 'Access', and 'Settings'. The 'Overview' tab is selected.

Installed add-ons \$0.00/month [Configure Add-ons](#)

- Heroku Postgres Hobby Dev postgresql-angular-72276

Dyno formation \$0.00/month [Configure Dynos](#)

This app is using **free** dynos

web gunicorn pocserver.wsgi --log-file - **ON**

Latest activity [All Activity](#)

- bodasmaresh@yahoo.com:** Deployed Nov 6 at 6:23 PM · v6
- bodasmaresh@yahoo.com:** Build succeeded Nov 6 at 6:21 PM · [View build log](#)
- bodasmaresh@yahoo.com:** @ref:postgresql-angular-72276 completed provisioning, setting DATABASE_URL. Nov 6 at 6:21 PM · v5
- bodasmaresh@yahoo.com:** Attach DATABASE (@ref:postgresql-angular-72276)

API response when not Logged In

← → ↻ <https://graphql-drf-poc-api-master.herokuapp.com/admingq/graphql/> ☆ 🚫 📱 Ⓜ | Paused 👤 ⋮

Django REST framework

Drf Admin Only Graph Ql

Drf Admin Only Graph Ql

GET ▾

GET /admingq/graphql/

HTTP 401 Unauthorized
Allow: OPTIONS, GET, POST
Content-Type: application/json
Vary: Accept
WWW-Authenticate: Token

```
{
  "detail": "Authentication credentials were not provided."
}
```


Authenticated User can run queries

The screenshot displays the GraphQL IDE interface. On the left, the query editor shows two queries: `getAllRisksByRisktype` and `getRisksByRisktype`. The first query is selected, and its variables are defined in the 'QUERY VARIABLES' section. The right pane shows the JSON response for the first query, which includes a list of risk instances and their associated risk types. The 'AdminQuery' sidebar on the far right lists the available schema types and their fields.

```
query getAllRisksByRisktype($risktypeid: Int!) {
  riskinstances: all_risks(risktype: $risktypeid) {
    id
    risktype
    risk_type_name
    risk_name
    risk_description
    risk_riskfields {
      id
      risktypefield
      risk
      risk_field_value
      risk_type_field_name
    }
  }
}
```

QUERY VARIABLES

```
{ "riskname": "Auto10", "first": 100, "after": null }
```

```
{
  "data": {
    "riskinstances": {
      "pageInfo": {
        "startCursor": "YXJyYX1jb25uZWNoaW9uOjA=",
        "endCursor": "YXJyYX1jb25uZWNoaW9uOjg=",
        "hasNextPage": false,
        "hasPreviousPage": false
      },
      "edges": [
        {
          "cursor": "YXJyYX1jb25uZWNoaW9uOjA=",
          "node": {
            "id": 663,
            "risktype": 1,
            "risk_type_name": "Automobile",
            "risk_name": "Auto100804",
            "risk_description": "Auto100804 Risk policy"
          }
        },
        {
          "cursor": "YXJyYX1jb25uZWNoaW9uOjE=",
          "node": {
            "id": 863,
            "risktype": 1,
            "risk_type_name": "Automobile",
            "risk_name": "Auto101421",
            "risk_description": "Auto101421 Risk policy"
          }
        },
        {
          "cursor": "YXJyYX1jb25uZWNoaW9uOjI=",
          "node": {
            "id": 6,
            "risktype": 1,
            "risk_type_name": "Automobile",
            "risk_name": "Auto104594",
            "risk_description": "Auto104594 Risk policy"
          }
        }
      ]
    }
  }
}
```

Schema AdminQuery

- risktype(
 id: Int
 risk_type_name: String
 risk_type_description: String
): RiskTypeQL
- risktypefield: RiskTypeFieldQL
- risk(
 id: Int
 risk_name: String
 risk_description: String
 risktype: Int
): RiskQL
- all_users: [UserQL]
- all_risks(risktype: Int): [RiskQL]
- risks(
 risktype: Int
 risk_name: String
 before: String
 after: String
 first: Int
 last: Int
): Risk_Connection
- all_risktypes: [RiskTypeQL]
- _debug: DjangoDebug

Query to retrieve all RiskTypes

```
query {  
  all_risktypes {  
    id,  
    risk_type_name,  
    risk_type_description,  
    risktype_risktypefields {  
      id,  
      risktype,  
      risk_type_field_name,  
      risk_type_field_enum,  
      risk_type_field_description  
    }  
  }  
}
```

Query to get all RiskTypes

```
1 query {  
2   all_risktypes{  
3     id,  
4     risk_type_name,  
5     risk_type_description,  
6     risktype_risktypefields {  
7       id,  
8       risktype,  
9       risk_type_field_name,  
10      risk_type_field_enum,  
11      risk_type_field_description  
12    }  
13  }  
14 }
```

QUERY VARIABLES

1 |

```
{  
  "data": {  
    "all_risktypes": [  
      {  
        "id": "1",  
        "risk_type_name": "Automobile",  
        "risk_type_description": "Type for Automobile Risk",  
        "risktype_risktypefields": [  
          {  
            "id": "1",  
            "risktype": "1",  
            "risk_type_field_name": "age",  
            "risk_type_field_enum": "integer",  
            "risk_type_field_description": "Age"  
          },  
          {  
            "id": "2",  
            "risktype": "1",  
            "risk_type_field_name": "policy_number",  
            "risk_type_field_enum": "integer",  
            "risk_type_field_description": "Policy Number"  
          },  
          {  
            "id": "3",  
            "risktype": "1",  
            "risk_type_field_name": "policy_bind_date",  
            "risk_type_field_enum": "date",  
            "risk_type_field_description": "Policy Date"  
          },  
          {  
            "id": "4",  
            "risktype": "1",  
            "risk_type_field_name": "policy_state",  
            "risk_type_field_enum": "text",  
            "risk_type_field_description": "Policy State"  
          }  
        ]  
      }  
    ]  
  }  
}
```

Query to Single RiskType

```
query getSingleRisktype($risktypeid:Int!) {  
  risktypeobj:risktype(id:$risktypeid){  
    id,  
    risk_type_name,  
    risk_type_description,  
    risktype_risktypefields {  
      id,  
      risktype,  
      risk_type_field_name,  
      risk_type_field_enum,  
      risk_type_field_description  
    }  
  }  
}
```

Get Single RiskType

localhost:9527/admingq/graphql/#query=query%20getSingleRisktype(%24risktypeid%3AInt!)%2...

GraphiQL

```
1 query getSingleRisktype($risktypeid:Int!) {
2   risktypeobj:risktype(id:$risktypeid){
3     id,
4     risk_type_name,
5     risk_type_description,
6     risktype_risktypefields {
7       id,
8       risktype,
9       risk_type_field_name,
10      risk_type_field_enum,
11      risk_type_field_description
12    }
13  }
14 }
```

QUERY VARIABLES

```
1 {
2   "risktypeid": 1
3 }
```

```
{
  "data": {
    "risktypeobj": {
      "id": "1",
      "risk_type_name": "Automobile",
      "risk_type_description": "Type for Automobile Risk",
      "risktype_risktypefields": [
        {
          "id": "1",
          "risktype": "1",
          "risk_type_field_name": "age",
          "risk_type_field_enum": "integer",
          "risk_type_field_description": "Age"
        },
        {
          "id": "2",
          "risktype": "1",
          "risk_type_field_name": "policy_number",
          "risk_type_field_enum": "integer",
          "risk_type_field_description": "Policy Number"
        },
        {
          "id": "3",
          "risktype": "1",
          "risk_type_field_name": "policy_bind_date",
          "risk_type_field_enum": "date",
          "risk_type_field_description": "Policy Date"
        },
        {
          "id": "4",
          "risktype": "1",
          "risk_type_field_name": "policy_state",
          "risk_type_field_enum": "text",
          "risk_type_field_description": "Policy State"
        },
        {
          "id": "5",
          "risktype": "1"
        }
      ]
    }
  }
}
```

Query to Single Risk

```
query getSingleRisk($riskid:Int!) {  
  riskinstance:risk(id:$riskid) {  
    id,  
    risktype,  
    risk_type_name,  
    risk_name,  
    risk_description,  
    risk_riskfields {  
      id,  
      risktypefield,  
      risk,  
      risk_type_field_enum,  
      risk_field_value,  
      risk_type_field_name  
    }  
  }  
}
```

Get Single Risk

localhost:9527/admingq/graphql/#query=query%20getSingleRisk(%24riskid%3AInt!)%20%7B%0...

GraphiQL

```
1 query getSingleRisk($riskid:Int!) {
2   riskinstance:risk(id:$riskid) {
3     id,
4     risktype,
5     risk_type_name,
6     risk_name,
7     risk_description,
8     risk_riskfields {
9       id,
10      risktypefield,
11      risk,
12      risk_type_field_enum,
13      risk_field_value,
14      risk_type_field_name
15    }
16  }
17 }
```

QUERY VARIABLES









```
1 {
2   "riskid": 1
3 }
4 }
```


```
{
  "data": {
    "riskinstance": {
      "id": "1",
      "risktype": 1,
      "risk_type_name": "Automobile",
      "risk_name": "Auto521585",
      "risk_description": "Auto521585 Risk policy",
      "risk_riskfields": [
        {
          "id": "1",
          "risktypefield": "1",
          "risk": "1",
          "risk_type_field_enum": "integer",
          "risk_field_value": "48",
          "risk_type_field_name": "age"
        },
        {
          "id": "2",
          "risktypefield": "2",
          "risk": "1",
          "risk_type_field_enum": "integer",
          "risk_field_value": "521585",
          "risk_type_field_name": "policy_number"
        },
        {
          "id": "3",
          "risktypefield": "3",
          "risk": "1",
          "risk_type_field_enum": "date",
          "risk_field_value": "10/17/2014",
          "risk_type_field_name": "policy_bind_date"
        },
        {
          "id": "4",
          "risktypefield": "4",
          "risk": "1",
          "risk_type_field_enum": "text"
        }
      ]
    }
  }
}
```

Fetch All RiskTypeKeys

```
query{  
  risk_type_keys: all_risktypes {  
    id  
    risk_type_name  
  }  
}
```


Get All RiskTypeKeys

localhost:9527/admingq/graphql/#operationName=undefined&query=query%7B%0A%20%20ri...       | Paused  

GraphQL  Prettify Merge History [< Docs](#docs)

```
1 query{
2   risk_type_keys: all_risktypes {
3     id
4     risk_type_name
5   }
6 }
```

```
{
  "data": {
    "risk_type_keys": [
      {
        "id": "1",
        "risk_type_name": "Automobile"
      },
      {
        "id": "4",
        "risk_type_name": "Home"
      }
    ]
  }
}
```

QUERY VARIABLES

1

Get Risks for Risk type page by page

```
query getRisksByRisktype($risktypeid: Int!, $first: Int, $after: String ) {  
  riskinstances:risks(risktype:$risktypeid, first:$first, after:$after){  
    pageInfo {  
      startCursor  
      endCursor  
      hasNextPage  
      hasPreviousPage  
    }  
    edges {  
      cursor  
      node {  
        id,  
        risk_name,  
        risk_description,  
        risk_riskfields {  
          id,  
          risktypefield,  
          risk,  
          risk_field_value,  
          risk_type_field_enum,  
          risk_type_field_name,  
          risk_type_field_description  
        }  
      }  
    }  
  }  
}
```

Get one page of Risks for RiskType

localhost:9527/admingq/graphql/#operationName=getRisksByRisktype&query=query%20getRis... Paused

GraphiQL Prettify Merge History < Docs

```
1 query getRisksByRisktype($risktypeid: Int!, $first: Int, $after: String) {
2   riskinstances: risks(risktypeid: $risktypeid, first: $first, after: $after) {
3     pageInfo {
4       startCursor
5       endCursor
6       hasNextPage
7       hasPreviousPage
8     }
9     edges {
10      cursor
11      node {
12        id
13        risk_name
14        risk_description
15        risk_riskfields {
16          id
17          risktypeid
18          risk
19          risk_field_value
20          risk_type_field_enum
21          risk_type_field_name
22          risk_type_field_description
23        }
24      }
25    }
26  }
27 }
```

QUERY VARIABLES

```
1 {"risktypeid": 4, "first": 1, "after": null}
```

```
{
  "data": {
    "riskinstances": {
      "pageInfo": {
        "startCursor": "YXJyYXljb25uZWN0aW9uOjA=",
        "endCursor": "YXJyYXljb25uZWN0aW9uOjA=",
        "hasNextPage": true,
        "hasPreviousPage": false
      },
      "edges": [
        {
          "cursor": "YXJyYXljb25uZWN0aW9uOjA=",
          "node": {
            "id": "Umlza190b2Rl0jIxNTM=",
            "risk_name": "Home12160159",
            "risk_description": "Home12160159 Risk policy",
            "risk_riskfields": [
              {
                "id": "36889",
                "risktypeid": "43",
                "risk": "2153",
                "risk_field_value": "1000",
                "risk_type_field_enum": "integer",
                "risk_type_field_name": "area",
                "risk_type_field_description": "Area"
              },
              {
                "id": "36890",
                "risktypeid": "44",
                "risk": "2153",
                "risk_field_value": "2/12/2017",
                "risk_type_field_enum": "date",
                "risk_type_field_name": "policy_bind_date",
                "risk_type_field_description": "Policy Date"
              },
              {
                "id": "36891",
                "risktypeid": "45"
              }
            ]
          }
        }
      ]
    }
  }
}
```

Create RiskType mutation

```
mutation createRiskType($riskTypeInput:RiskTypeInput!) {  
  create_risktype(input:$riskTypeInput) {  
    ok,  
    risktype{  
      risk_type_name,  
      risk_type_description,  
      risktype_risktypefields {  
        risk_type_field_name,  
        risk_type_field_enum,  
        risk_type_field_description  
      }  
    }  
  }  
}
```

Create RiskType mutation

← → ↻ ⓘ localhost:9527/admingq/graphql/#operationName=getRisksByRisktype&query=query%20getRis... 🔍 ☆ ⚙️ ▼ 🧠 Ⓐ | Paused 👤 ⋮

GraphiQL ▶ Prettify Merge History < Docs

```
1 query getRisksByRisktype($risktypeid: Int!, $first: Int, $after: String) {
2   riskinstances: risks(risktypeid: $risktypeid, first: $first, after: $after) {
3     pageInfo {
4       startCursor
5       endCursor
6       hasNextPage
7       hasPreviousPage
8     }
9   }
10  edges {
11    cursor
12    node {
13      id
14      risk_name
15      risk_description
16      risk_riskfields {
17        id
18        risktypeid
19        risk
20        risk_field_value
21        risk_type_field_enum
22        risk_type_field_name
23        risk_type_field_description
24      }
25    }
26  }
27 }
```

QUERY VARIABLES

```
1 {"risktypeid": 4, "first": 1, "after": null}
```

```
{
  "data": {
    "riskinstances": {
      "pageInfo": {
        "startCursor": "YXJyYXljb25uZWN0aW9uOjA=",
        "endCursor": "YXJyYXljb25uZWN0aW9uOjA=",
        "hasNextPage": true,
        "hasPreviousPage": false
      },
      "edges": [
        {
          "cursor": "YXJyYXljb25uZWN0aW9uOjA=",
          "node": {
            "id": "Umlza190b2Rl0jIxNTM=",
            "risk_name": "Home12160159",
            "risk_description": "Home12160159 Risk policy",
            "risk_riskfields": [
              {
                "id": "36889",
                "risktypeid": "43",
                "risk": "2153",
                "risk_field_value": "1000",
                "risk_type_field_enum": "integer",
                "risk_type_field_name": "area",
                "risk_type_field_description": "Area"
              },
              {
                "id": "36890",
                "risktypeid": "44",
                "risk": "2153",
                "risk_field_value": "2/12/2017",
                "risk_type_field_enum": "date",
                "risk_type_field_name": "policy_bind_date",
                "risk_type_field_description": "Policy Date"
              },
              {
                "id": "36891",
                "risktypeid": "45"
              }
            ]
          }
        }
      ]
    }
  }
}
```

<https://django-poc-session-maheshbodas.herokuapp.com/docs/>

The screenshot shows a web browser displaying the API documentation for 'Pastebin API' at the URL <https://django-poc-session-maheshbodas.herokuapp.com/docs/>. The browser's address bar and navigation icons are visible at the top. On the left, a dark sidebar lists API endpoints: 'api-token-auth', 'auth', 'riskkeys', 'risks', 'risktypekeys', 'risktypes', and 'users', each with a dropdown arrow. Below these are links for 'Authentication' (with a 'session' tab) and 'Source Code' (with a 'shell' tab). The main content area features the title 'Pastebin API' and a description: 'A Web API for creating and viewing RiskTypes and RiskInstances based on that.' Below this, the 'api-token-auth' endpoint is detailed. It includes a 'create' section with a 'POST' method and a path '/api-token-auth/'. To the right of this is a green 'INTERACT' button. Further right is a code block with the following content:

```
# Install the command line client
$ pip install coreapi-cli
```

```
# Load the schema document
$ coreapi get https://django-poc-session-maheshbodas
```

```
# Interact with the API endpoint
$ coreapi action api-token-auth create
```

Below the 'create' section is the 'auth' section, followed by a 'login > create' section with another 'INTERACT' button and a code block:

```
# Load the schema document
$ coreapi get https://django-poc-session-maheshbodas
```

<https://django-poc-session-maheshbodas.herokuapp.com/schema>

The screenshot shows a web browser window with the address bar displaying `https://django-poc-session-maheshbodas.herokuapp.com/schema`. The page title is "Django REST framework" and the user is logged in as "mahesh.bodas". The breadcrumb "Api Root / Schema" is visible. The main heading is "Schema", with "OPTIONS" and "GET" buttons to its right. Below the heading, the request method "GET /schema/" is shown. The response details are as follows:

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/coreapi+json
Vary: Accept

```
{
  "_type": "document",
  "_meta": {
    "url": "https://django-poc-session-maheshbodas.herokuapp.com/schema/",
    "title": "Pastebin API"
  },
  "api-token-auth": {
    "create": {
      "_type": "link",
      "url": "/api-token-auth/",
      "action": "post"
    }
  }
}
```