**Zbooni OAuth WorkFlow**

Contents

[1. To obtain a valid access\_token first we must register an application. DOT has a set of customizable views you can use to CRUD application instances, just point your browser at: 1](#_Toc38816703)

[**2.** Registration: At this point we’re ready to request an access\_token. 2](#_Toc38816704)

[3. User Activation: Provide email ID and token shared to activate user 3](#_Toc38816705)

[4. Login: the endpoint should require email and password, and if correct, responds with an OAuth2 bearer token. 4](#_Toc38816706)

[5. Change Password 4](#_Toc38816707)

[6. Users list: 6](#_Toc38816708)

[7. Postman Collection : Collection attached 6](#_Toc38816709)

**Introduction:**

Django OAuth Toolkit is used in this project to used tokenization. Browser and API postman support provided to access the endpoints. The detailed steps to be followed are described below in sequence. Except the first registration all are bearer token based authentication.

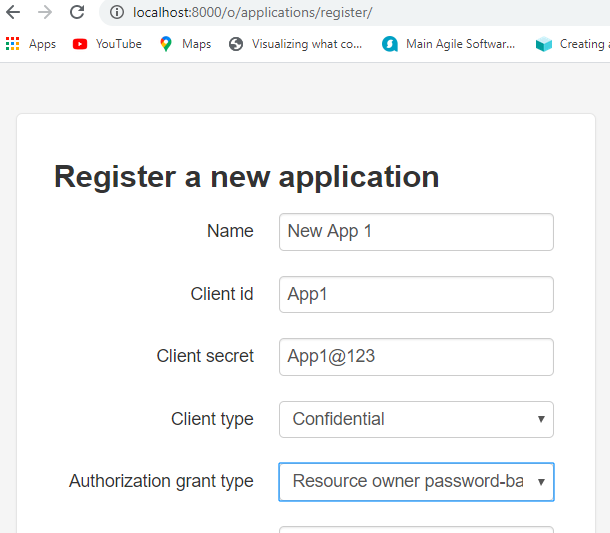
**Key Features:**

* Virtual Env used
* Custom User Model
* Django OAuth Toolkit
* API View
* Bearer token
* Activating user on receipt of token using login
* Well documented code
* User documentation provided for API

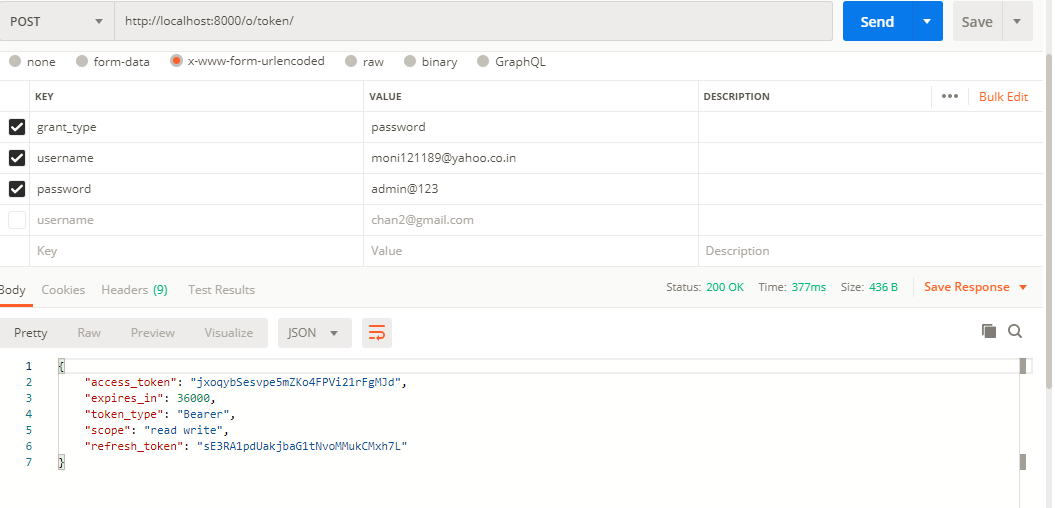
**Steps to be followed**

# To obtain a valid access\_token first we must register an application. DOT has a set of customizable views you can use to CRUD application instances, just point your browser at:

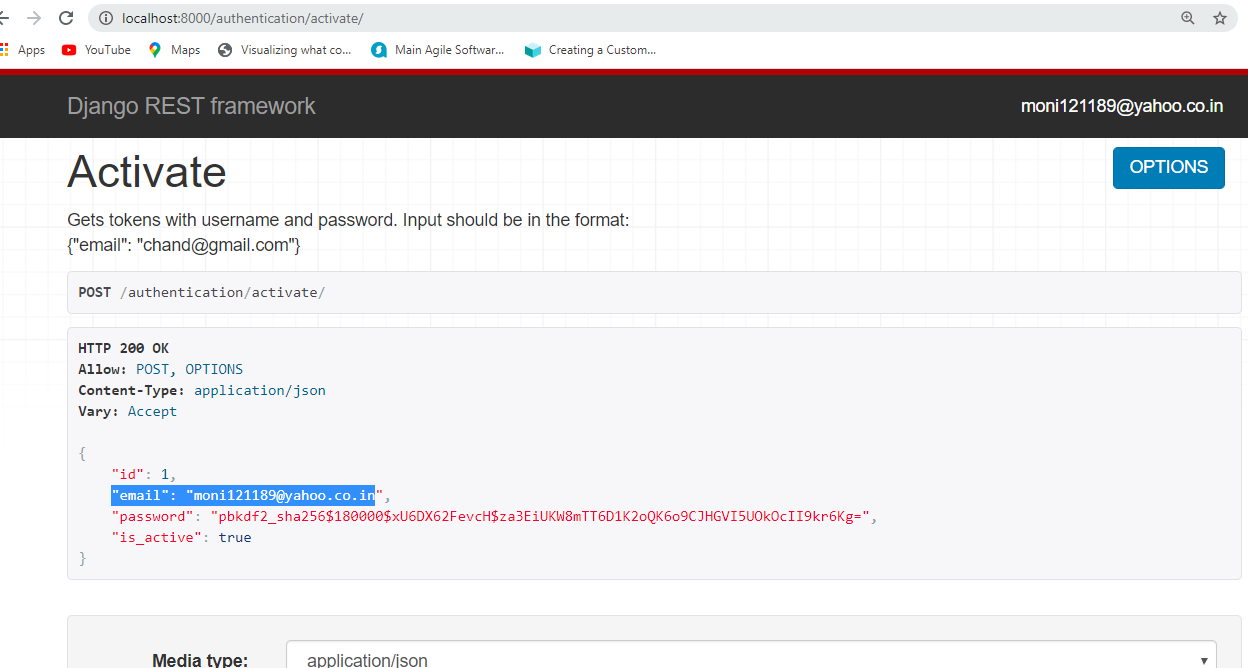
[**http://localhost:8000/o/applications/**](http://localhost:8000/o/applications/)

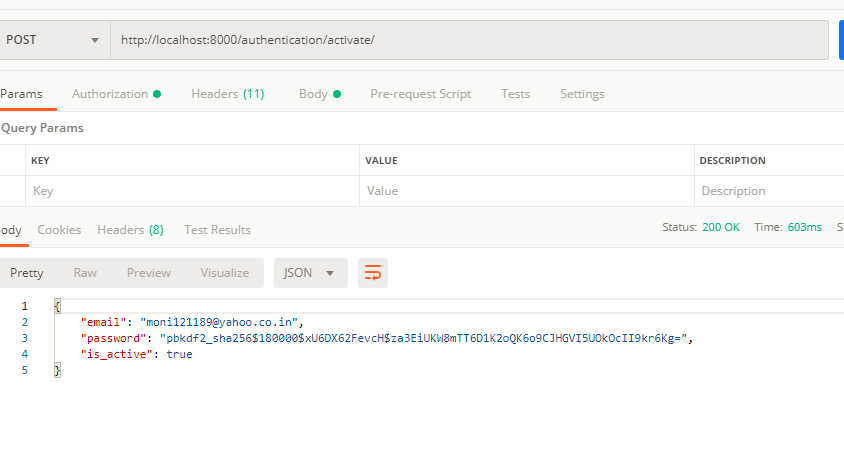


# Registration: At this point we’re ready to request an access\_token.

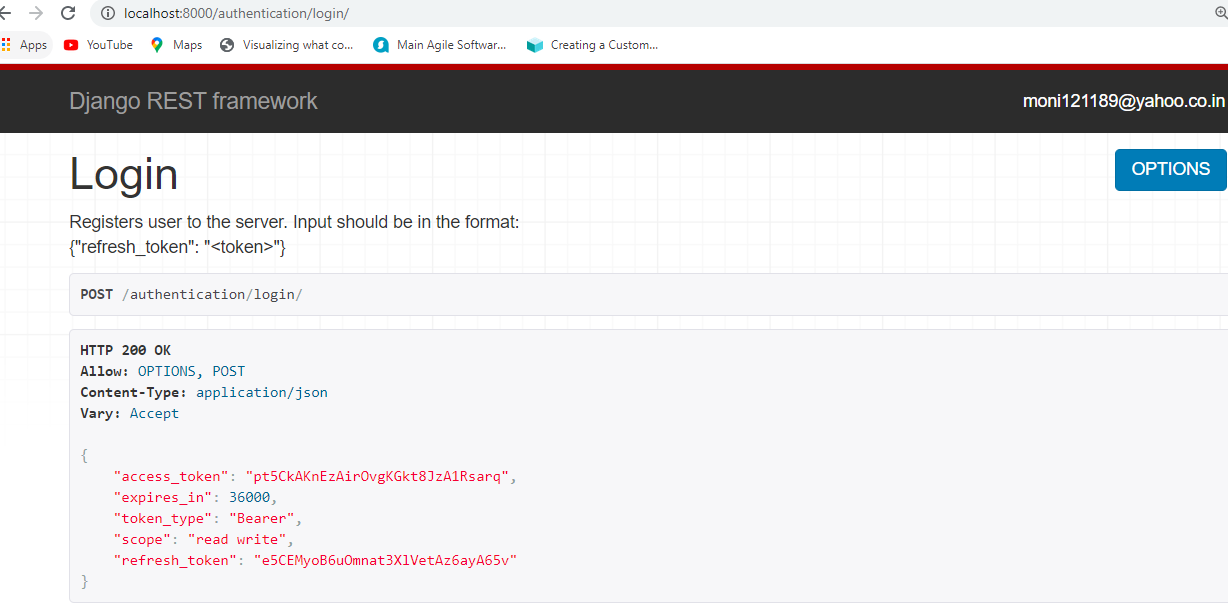


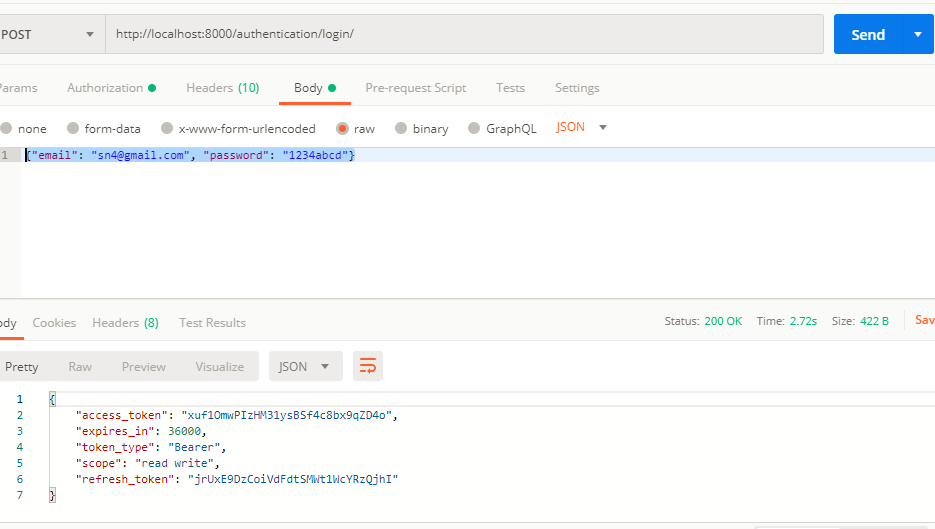
# User Activation: Provide email ID and token shared to activate user



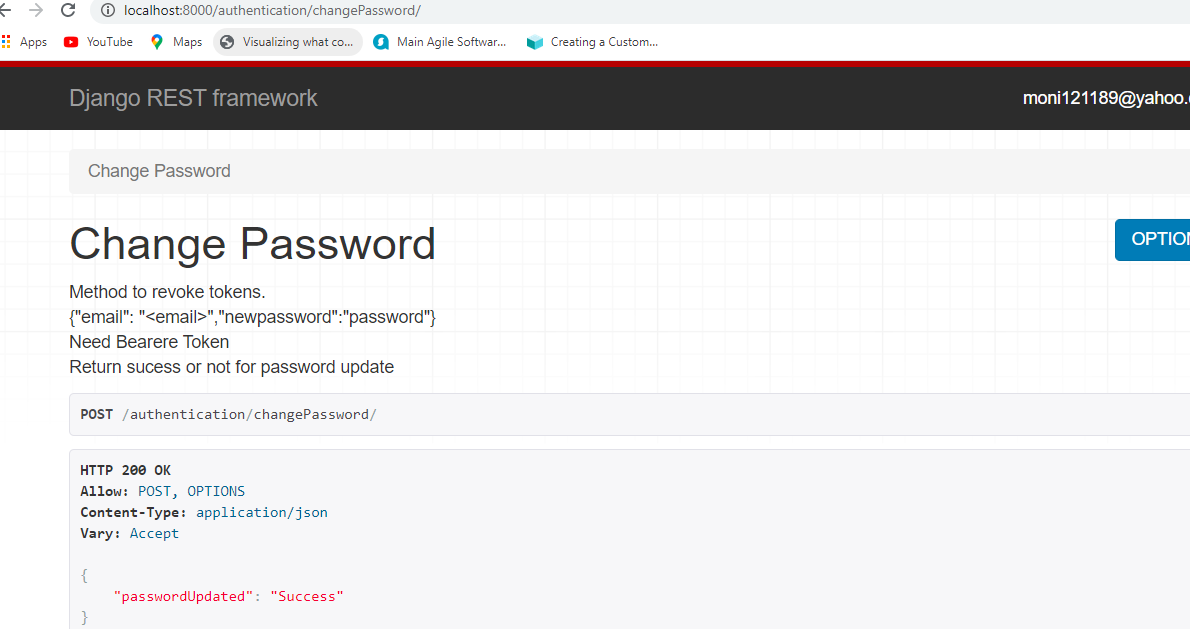


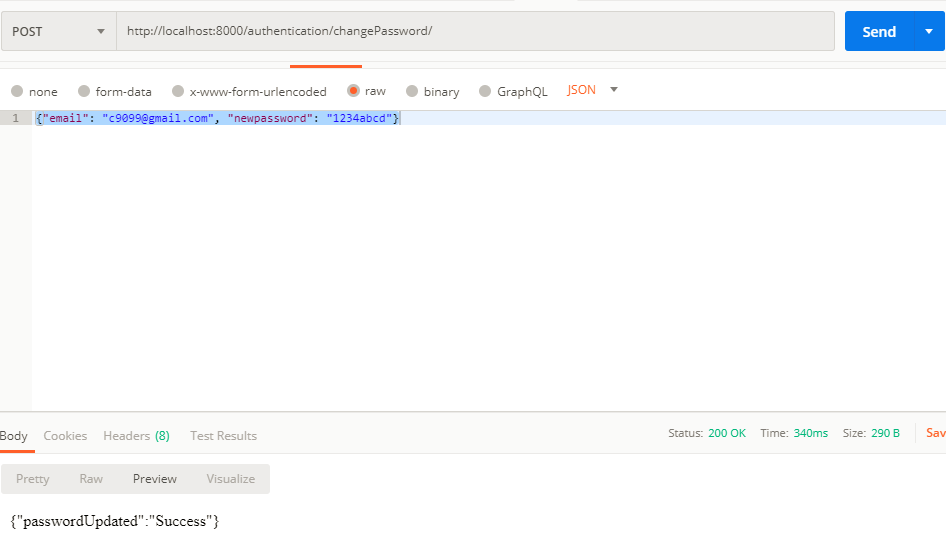
# Login: the endpoint should require email and password, and if correct, responds with an OAuth2 bearer token.



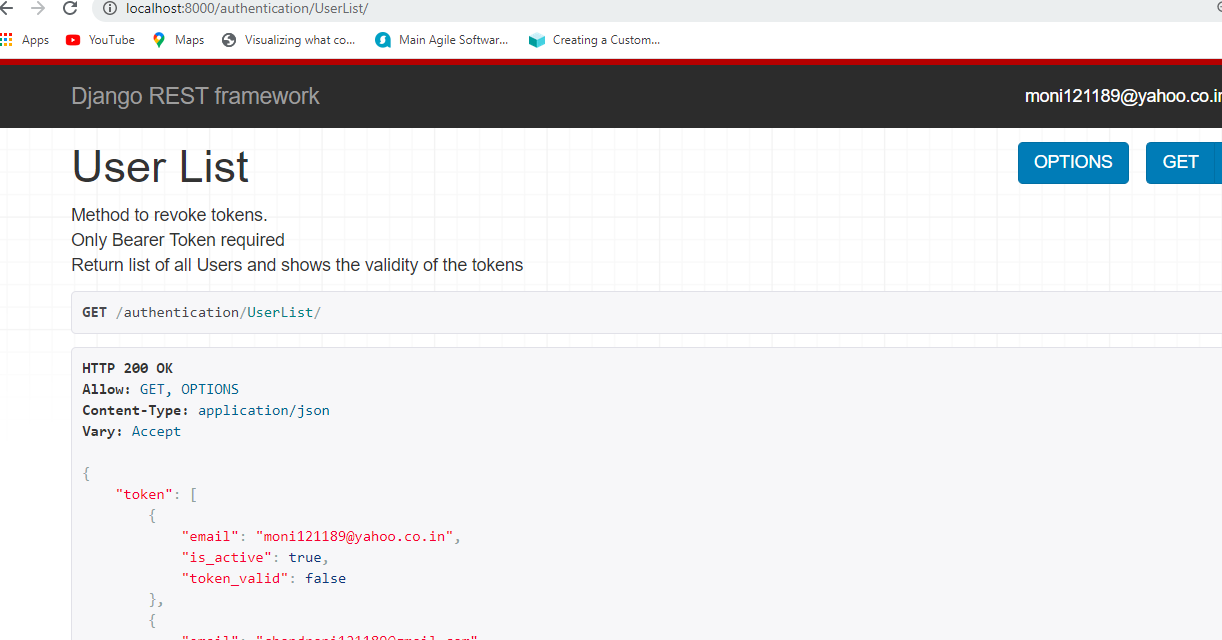


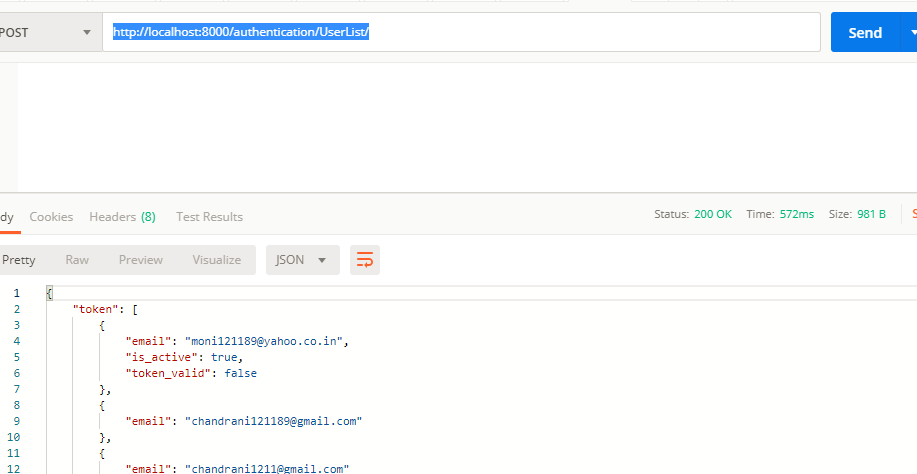
# Change Password





# Users list:





# Postman Collection : Collection attached

