**Shubhangi Agarwal**

**Release 1**

**Status**

Implemented work

* Basic database model for taking user input (Through Django, sqlite3)
* Handling http\_requests and responses (Through Django)
* Addition of user input into training data (Through Django)
* A REST API for connecting with frontend (Through Django)
* API mapping to create a request from React to Django (Through React.js)

In Progress

* Connection of frontend with backend for getting an end-to-end workflow

**Backend Details**

Database Schema

* Partner : {

id : AutoField, primary key

firstName : CharField

lastName : CharField

email : EmailField

contact : CharField

address : CharField

city : CharField

state : CharField

typestore : CharField

size : CharField

workingemployees : PositiveIntegerField

customers : PositiveIntegerField

service : CharField

password : CharField

confirmPassword : CharField

}

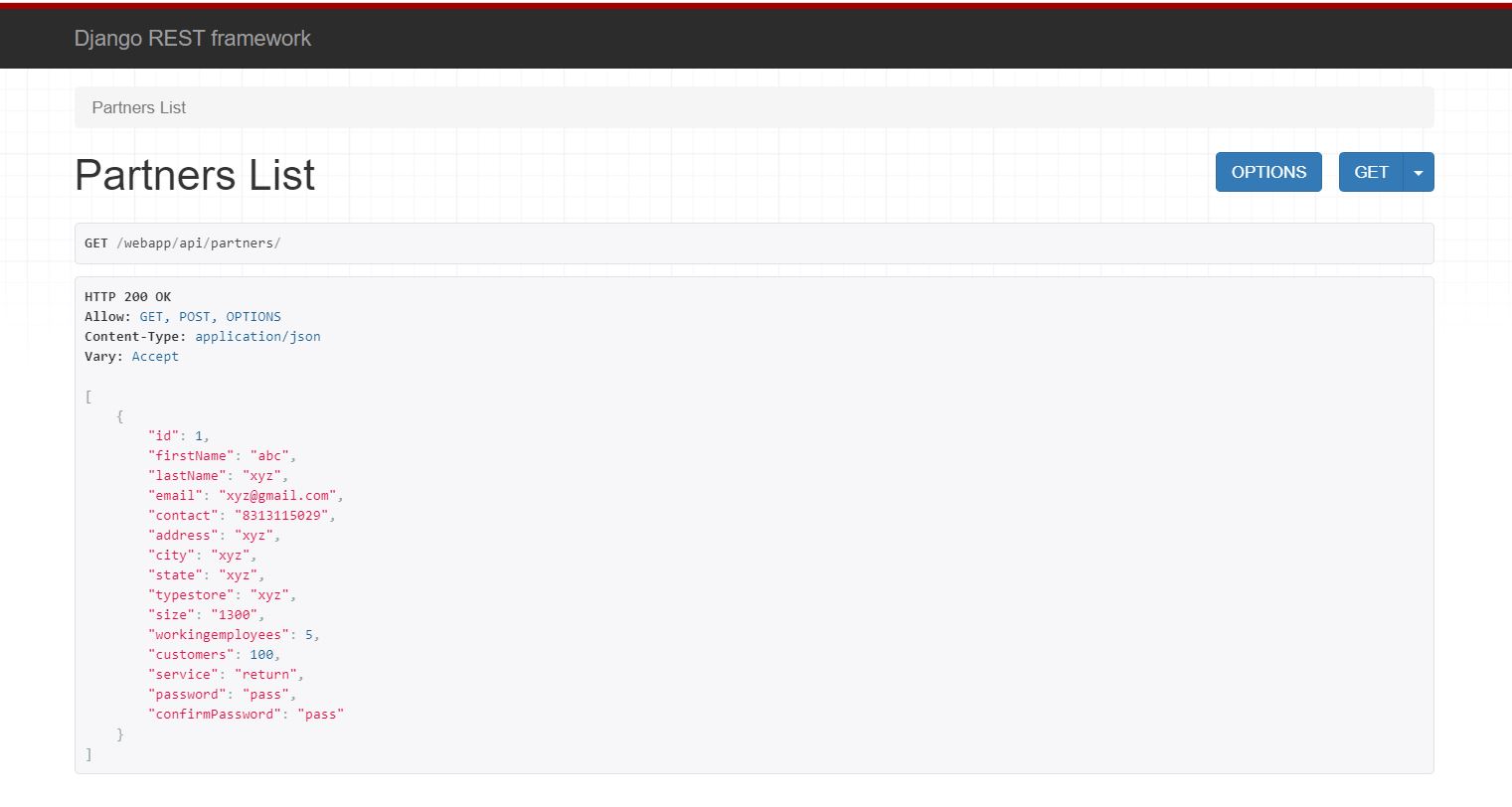
Routes

* webapp/api/partners/list/ : { method : ‘post’ , ‘get’ }
* webapp/api/partners/list/details/ : { method : ‘put’ , ‘delete’ }

Requests and responses

**{ ‘request\_method’ ? response\_if\_true : response\_if\_false }**

* ‘get’ ? data stored in ‘Partner’ database : HTTP\_400\_BAD\_REQUEST
* ‘post’ ? HTTP\_201\_CREATED : HTTP\_400\_BAD\_REQUEST
* ‘put’ ? HTTP\_204\_NO\_CONTENT : HTTP\_400\_BAD\_REQUEST
* ‘delete’ ? HTTP\_204\_NO\_CONTENT : HTTP\_400\_BAD\_REQUEST



**Fig. 1:** Response for ‘get’ request

**Technical Challenges**

* Understanding MVC and MVT architecture : Solved
* Database creation through Django : Solved
* Sending requests from React app to Django : In Progress