# Bondora Invest Environment Guide (EC2 Ubuntu Web Server)

References:

# https://medium.com/@charlesthk/deploy-nginx-django-uwsgi-on-aws-ec2-amazon-linux-517a683163c6

https://medium.freecodecamp.org/django-uwsgi-nginx-postgresql-setup-on-aws-ec2-ubuntu16-04-with-python-3-6-6c58698ae9d3

https://www.digitalocean.com/community/tutorials/how-to-serve-django-applications-with-uwsgi-and-nginx-on-ubuntu-16-04#setting-up-the-uwsgi-application-server

https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-with-let-s-encrypt-on-ubuntu-16-04

## Table of Contents

# 1 Local Django project to GitHub

## 1.1 GitHub

```
Create repository on GitHub:   consumer_lending
```

## 1.2 Actions to upload Django project to GitHub

```
mkdir consumer_lending

echo "# consumer_lending" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/KimmoOjala/consumer_lending.git
git push -u origin master

# Copy Django project to local consumer_lending directory
# Push project to GitHub

git add .
git commit -m "Second commit"
git push -u origin master
```

# 2 Bondora

# Create Bondora app

https://api.bondora.com/Application/Create

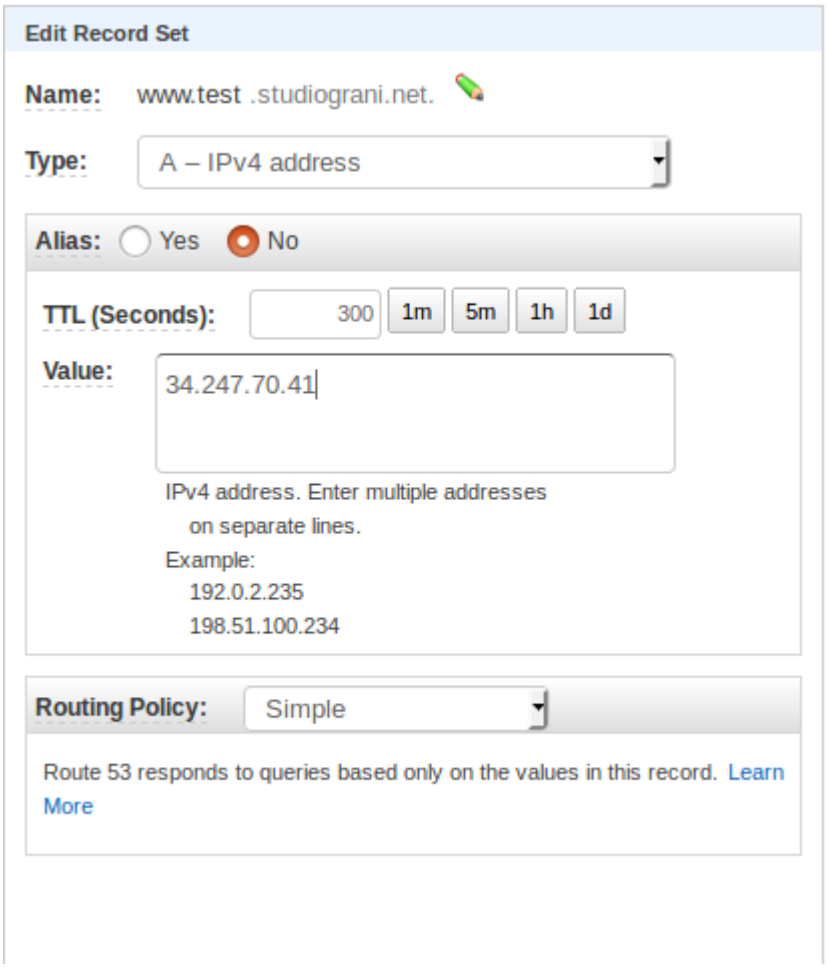
| | |
|---|---|
| Application Name: | reporting_app |
| Homepage URL: | https://www.reporting.studiograni.net/authorization/start |
| Application description: | reporting app |
| Authorizaton callback URL: | https://www.reporting.studiograni.net/authorization/end |

# 3 AWS Environment

## 3.1 Route 53

Create record in hosted zone studiograni.net.

Value of record is IP address of AWS EC2 Ubuntu Server Instance (Web Server):




Note: check for the new public IP each time you restart the server.

## 3.2 EC2 Instance (Web Server)

### 3.2.1 Launch ECS instance

Launch Ubuntu Server instance (freetier)

Auto-assign Public IP: Enable

Security Group: werbserverSG

### 3.2.2 ssh to instance and get updates

ssh -i investKeyPair.pem ubuntu@ec2-34-245-182-5.eu-west-1.compute.amazonaws.com
Note: check for the new public IP each time you restart the server

```
sudo apt-get update && sudo apt-get upgrade -y
```

### 3.2.3 Install Python 3.6 and dependencies

```
sudo apt install build-essential checkinstall
```

```
sudo apt install libreadline-gplv2-dev libncursesw5-dev libssl-dev libsqlite3-
dev tk-dev libgdbm-dev libc6-dev libbz2-dev


cd /opt && sudo wget https://www.python.org/ftp/python/3.6.6/Python-3.6.6.tar.xz

sudo tar -xvf Python-3.6.6.tar.xz

cd Python-3.6.6/

sudo ./configure

sudo make && sudo make install

sudo rm -rf Python-3.6.6.tar.xz



# Check python version

python3 -V
```

### 3.2.4 Installing and configuring PostgresSQL

```
sudo apt install postgresql postgresql-contrib

sudo su - postgres

psql

postgres=# CREATE DATABASE bondora_db;

Changing default password for postgres while in psql terminal
```

**postgres=# \password**
<new_password>

```
\q
```

### 3.2.5 Installing Git and pulling your code from git repo

```
sudo apt-get install git

cd /home/ubuntu

git clone https://github.com/KimmoOjala/consumer_lending

To update cloned repository:
git pull
```

### 3.2.6 Creating virtual environment using Python 3.6 in current directory

```
cd /home/ubuntu
```

python3.6 -m venv ve

source ve/bin/activate

```
cd consumer_lending

pip install -r requirements.txt
```

### 3.2.7 Django project commands

```
cd bondora_invest

python manage.py migrate

python manage.py createsuperuser

      superuser: ubuntu, password: dejah_thorris

python manage.py collectstatic

For testing:
python manage.py runserver 0.0.0.0:8000

http://www.test.studiograni.net:8000
```

### 3.2.8 Setting up the uWSGI Application Server

```
sudo pip3 install uwsgi


Test:
uwsgi --http :8080 --home /home/ubuntu/ve --chdir
/home/ubuntu/consumer_lending/bondora_invest/ -w firstsite.wsgi

Here, we've told uWSGI to use our virtual environment located in our
/home/ubuntu/ve directory, to change to our project's directory, and to use the
wsgi.py file stored within our inner firstsite directory to serve the file
(using the firstsite.wsgi Python module syntax). For our test, we told it to
serve HTTP on port 8080.
```

### 3.2.9 Creating Configuration Files

Running uWSGI from the command line is useful for testing, but isn't particularly helpful for an actual deployment. Instead, we will run uWSGI in "Emperor mode", which allows a master process to manage separate applications automatically given a set of configuration files.

Create a directory that will hold your configuration files. Since this is a global process, we will create a directory called `/etc/uwsgi/sites` to store our configuration files:

```
sudo mkdir -p /etc/uwsgi/sites
```

```
sudo nano /etc/uwsgi/sites/bondora_invest.ini

# Copy the below code to bondora_invest.ini

[uwsgi]
project = bondora_invest
uid = ubuntu
base = /home/%(uid)

chdir = /home/ubuntu/consumer_lending/%(project)
virtualenv = /home/ubuntu/ve/

module = %(project).wsgi:application

master = true
processes = 5

socket = /run/uwsgi/%(project).sock
chown-socket = %(uid):www-data
chmod-socket = 660
vacuum = true
```

### 3.2.10 Create a systemd Unit File for uWSGI

```
sudo nano /etc/systemd/system/uwsgi.service

# Copy the below code to uwsgi.service

[Unit]
Description=uWSGI Emperor service

[Service]
ExecStartPre=/bin/bash -c 'mkdir -p /run/uwsgi; chown ubuntu:www-data
/run/uwsgi'
ExecStart=/usr/local/bin/uwsgi --emperor /etc/uwsgi/sites
Restart=always
KillSignal=SIGQUIT
Type=notify
NotifyAccess=all

[Install]
WantedBy=multi-user.target
```

# Reload systemctl daemon to reload systemd manager configuration and recreate the entire dependency tree

```
$ sudo systemctl daemon-reload
```

# Enable uwsgi service to start on system reboot

```
$ sudo systemctl enable uwsgi
```

# Start uwsgi service

```
$ sudo service uwsgi start
```

# Restart uwsgi service

```
$ sudo service uwsgi restart
```

# Check uwsgi service status

```
$ sudo service uwsgi status
```

### 3.2.11 Setting Up NGINX on EC2 for uWSGI

```
sudo apt-get install nginx

sudo nano /etc/nginx/sites-available/bondora_invest

# Copy the below code to bondora_invest

server {
    listen 80;
    server_name www.test.studiograni.net;

    location = /favicon.ico { access_log off; log_not_found off; }
    location /static/ {
        root /home/ubuntu/consumer_lending/bondora_invest;
    }

    location / {
        include         uwsgi_params;
        uwsgi_pass      unix:/run/uwsgi/bondora_invest.sock;
    }
}
```

# Next, link your new configuration file to Nginx's sites-enabled directory to enable it:

```
sudo ln -s /etc/nginx/sites-available/bondora_invest /etc/nginx/sites-enabled
```

# Check the configuration syntax by typing:

```
sudo nginx -t
```

# If no syntax errors are detected, you can restart your Nginx service to load the new configuration:

```
sudo systemctl restart nginx

sudo systemctl enable nginx
```

### 3.2.12 Installing Certbot

```
sudo add-apt-repository ppa:certbot/certbot

sudo apt-get update

sudo apt-get install python-certbot-nginx
```

### 3.2.13 Allowing HTTPS through the firewall

```
Inbound:
```

| Type | Protocol | Port Range | Source | Description |
|---|---|---|---|---|
| HTTP | TCP | 80 | 0.0.0.0/0 | |
| HTTP | TCP | 80 | ::/0 | |
| All TCP | TCP | 0 - 65535 | 82.203.157.33/32 | |
| SSH | TCP | 22 | 82.203.157.33/32 | |
| HTTPS | TCP | 443 | 0.0.0.0/0 | |
| HTTPS | TCP | 443 | ::/0 | |

Https needs to remain open to the world so that certbot can verify connections.

### 3.2.14 Obtaining an SSL Certificate

```
sudo certbot --nginx -d www.reporting.studiograni.net
```

test: https://www.reporting.studiograni.net/

### 3.2.15 Verifying Certbot Auto-Renewal

Let's Encrypt's certificates are only valid for ninety days. This is to encourage users to automate their certificate renewal process. The certbot package we installed takes care of this for us by running ‘certbot renew’ twice a day via a systemd timer. On non-systemd distributions this functionality is provided by a script placed in /etc/cron.d. This task runs twice a day and will renew any certificate that's within thirty days of expiration.

To test the renewal process, you can do a dry run with certbot:

```
sudo certbot renew --dry-run
```

## 3.3 RDS

### 3.3.1 Launch RDS PostGreSQL instance and create db

```
Engine: PostgreSQL
```

Only enable options eligible for RDS Free Usage Tier

```
DB instance identifier:  bondora-db-instance
username: ubuntu
pasword: <PASSWORD>
VPC: <vpc_code>
Public accessibility: No
Create new VPC security group:
        •  Name: bondora_db_SG

Database name: bondora_db
```

### 3.3.2 Security Group of PostgreSQL instance

Inbound

| Type | Protocol | Port Range | Source | Description |
|---|---|---|---|---|
| PostgreSQL | TCP | 5432 | 0.0.0.0/0 | |
| PostgreSQL | TCP | 5432 | ::/0 | |
| SSH | TCP | 22 | 34.248.152.85/32 | |

Note: To allow for access to bonda-db-instance from Web Server (or Bastion Host) inbound PostgreSQL traffic needs to be open to the world (0.0.0.0/0), but the intance is not actually available in the internet, since Public accessibility was chosen as “No”.

## 3.4 Change Django project db (from db on Web Server) to RDS db

### 3.4.1 Edit settings.py

cd ~/consumer_lending
source ve/bin/activate

sudo nano ~/consumer_lending/bondora_invest/bondora_invest/settings.py

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'bondora_db',
        'USER': 'ubuntu',
        'PASSWORD': '<PASSWORD>',
        'HOST': '<HOST_DNS>',
        'PORT': '5432',
    }

}
```

### 3.4.2 Commands

```
cd ~/consumer_lending/bondora_invest

python manage.py migrate

python manage.py createsuperuser

      superuser: ubuntu, password: <PASSWORD>
```

# 4 Connect to AWS postgreSQL instance

## 4.1 Check inbound rules of Security Groups

Note: instead of a webserver you could also use a bastion server, which would be just another ubuntu instance, but with no other ports open than just port 22 for SSH traffic from your own IP address.

**webserverSG**
Used by: Web Server

Inbound

| Type | Protocol | Port Range | Source | Description |
|---|---|---|---|---|
| HTTP | TCP | 80 | 0.0.0.0/0 | |
| HTTP | TCP | 80 | ::/0 | |
| All TCP | TCP | 0 - 65535 | 82.203.157.33/32 | |
| SSH | TCP | 22 | 82.203.157.33/32 | |
| HTTPS | TCP | 443 | 0.0.0.0/0 | |
| HTTPS | TCP | 443 | ::/0 | |

Outbound

| Type | Protocol | Port Range | Destination | Description |
|---|---|---|---|---|
| All traffic | All | All | 0.0.0.0/0 | |

**Bondora_db_SG**
Used by: PostgreSQL database

Inbound

| Type | Protocol | Port Range | Source | Description |
|---|---|---|---|---|
| PostgreSQL | TCP | 5432 | 0.0.0.0/0 | |
| PostgreSQL | TCP | 5432 | ::/0 | |
| SSH | TCP | 22 | 34.248.152.85/32 | |

Outbound

| Type | Protocol | Port Range | Destination | Description |
|---|---|---|---|---|
| All traffic | All | All | 0.0.0.0/0 | |

## 4.2 Install pgadmin locally

https://linuxhint.com/install-pgadmin4-ubuntu/

mkdir pgAdmin4
cd pgAdmin4
python3.6 -m venv pgadmin_ve

source pgadmin_ve/bin/activate
(pgadmin_ve) kimmo@kimmo-ubuntu:~/pgAdmin4$

wget https://ftp.postgresql.org/pub/pgadmin/pgadmin4/v3.3/pip/pgadmin4-3.3-py2.py3-none-any.whl

pip install pgadmin4-3.3-py2.py3-none-any.whl

**Configure and run pgAdmin 4**

After completing the installation steps, you have to create a configuration file to run this software. Create a new file named **config_local.py** in lib/python2.7/site-packages/pgadmin4/ folder using nano editor.

nano lib/python2.7/site-packages/pgadmin4/config_local.py

# Add the following content in c**onfig_local.py**.

```
import os
DATA_DIR = os.path.realpath(os.path.expanduser(u'~/.pgadmin/'))
LOG_FILE = os.path.join(DATA_DIR, 'pgadmin4.log')
SQLITE_PATH = os.path.join(DATA_DIR, 'pgadmin4.db')
SESSION_DB_PATH = os.path.join(DATA_DIR, 'sessions')
STORAGE_DIR = os.path.join(DATA_DIR, 'storage')
SERVER_MODE = False
```

Now, use the following commands to run pgAdmin.

(pgadmin_ve) kimmo@kimmo-ubuntu:  cd pgadmin_ve
(pgadmin_ve) kimmo@kimmo-ubuntu: python lib/python3.6/site-packages/pgadmin4/pgAdmin4.py

## 4.3 Connect using local pgadmin client

sudo service postgresql stop

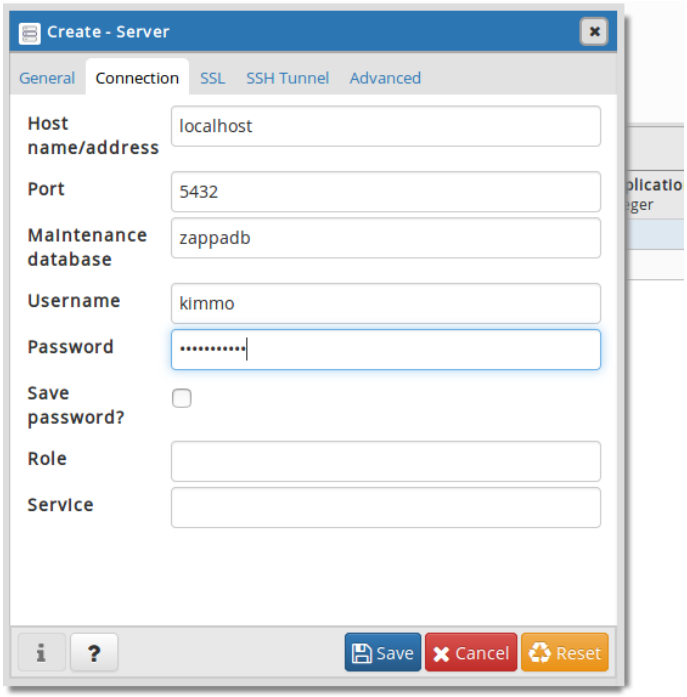
ssh -i investKeyPair.pem -l ubuntu -L 5432:<DB_DNS>:5432 34.245.182.5

Your local machine is now listening on port 5432 and will forward any of those connections to <Web-Server-ip> which in turn will forward it to port 5432 on <rds-private-ip>

**To test connection from webserver to database:**
ssh ubuntu@34.248.152.85 -i investKeyPair.pem

psql -h <DB_DNS> -p 5432 -d bondora_db -U ubuntu

In pgadmin create new server:




# 5 Database tables

```
CREATE TABLE authentication
(
	id SERIAL,
	access_token  character varying,
	CONSTRAINT authentication_pkey PRIMARY KEY (id)
)



CREATE TABLE public_report
(
	ActiveLateCategory character varying(10),
	ActiveScheduleFirstPaymentReached boolean,
	Age int,
	Amount int,
	AmountOfPreviousLoansBeforeLoan int,
	ApplicationSignedHour int,
	ApplicationSignedWeekday int,
	AppliedAmount int,
	BiddingStartedOn character varying(30),
	BidsApi int,
	BidsManual int,
	BidsPortfolioManager int,
	City character varying(100),
	ContractEndDate character varying(30),
	Country character(2),
	County character varying(100),
```

CreditScoreEeMini int,
CreditScoreEsEquifaxRisk character varying(30),
CreditScoreEsMicroL character varying(30),
CreditScoreFiAsiakasTietoRiskGrade character varying(30),
CurrentDebtDaysPrimary int,
CurrentDebtDaysSecondary int,
DateOfBirth character varying(30),
DebtOccuredOn character varying(30),
DebtOccuredOnForSecondary character varying(30),
DebtToIncome real,
DefaultDate character varying(30),
EAD1 character varying(30),
EAD2 character varying(30),
EL_V0 real,
EL_V1 real,
EL_V2 real,
Education int,
EmploymentDurationCurrentEmployer character varying(30),
EmploymentPosition character varying(200),
EmploymentStatus int,
ExistingLiabilities int,
ExpectedLoss real,
ExpectedReturn real,
FirstPaymentDate character varying(30),
FreeCash real,
Gender int,
GracePeriodEnd character varying(30),
GracePeriodStart character varying(30),
HomeOwnershipType int,
IncomeFromChildSupport int,
IncomeFromFamilyAllowance int,
IncomeFromLeavePay  int,
IncomeFromPension int,
IncomeFromPrincipalEmployer int,
IncomeFromSocialWelfare int,
IncomeOther int,
IncomeTotal int,
Interest real,
InterestAndPenaltyBalance real,
InterestAndPenaltyDebtServicingCost real,
InterestAndPenaltyPaymentsMade real,
InterestAndPenaltyWriteOffs real,
InterestRecovery real,
LanguageCode int,
LastPaymentOn character varying(30),
LiabilitiesTotal real,
ListedOnUTC character varying(30),
LoanApplicationStartedDate character varying(30),
LoanCancelled boolean,
LoanDate character varying(30),
LoanDuration int,
LoanId character varying(40) NOT NULL,

```
    LoanNumber int,
    LossGivenDefault real,
    MaritalStatus int,
    MaturityDate_Last character varying(30),
    MaturityDate_Original character varying(30),
    ModelVersion int,
    MonthlyPayment int,
    MonthlyPaymentDay int,
    NewCreditCustomer boolean,
    NextPaymentDate character varying(30),
    NextPaymentNr int,
    NoOfPreviousLoansBeforeLoan int,
    NrOfDependants int,
    NrOfScheduledPayments int,
    OccupationArea int,
    PlannedInterestPostDefault real,
    PlannedInterestTillDate real,
    PlannedPrincipalPostDefault real,
    PlannedPrincipalTillDate real,
    PreviousEarlyRepaymentsBeforeLoan real,
    PreviousEarlyRepaymentsCountBeforeLoan real,
    PreviousRepaymentsBeforeLoan real,
    PrincipalBalance real,
    PrincipalDebtServicingCost int,
    PrincipalOverdueBySchedule real,
    PrincipalPaymentsMade real,
    PrincipalRecovery real,
    PrincipalWriteOffs real,
    ProbabilityOfDefault real,
    Rating character varying(30),
    Rating_V0 character varying(30),
    Rating_V1 character varying(30),
    Rating_V2 character varying(30),
    ReScheduledOn character varying(30),
    RecoveryStage int,
    RefinanceLiabilities int,
    Restructured boolean,
    StageActiveSince character varying(30),
    Status character varying(30),
    UseOfLoan int,
    UserName character varying(30),
    VerificationType int,
    WorkExperience character varying(30),
    WorseLateCategory character varying(30)
)
```

# 6 Service and Maintenance

## 6.1 Actions to restart AWS environment

Actions to restart environment:

- Start web server (Ubuntu_webserver) → new IP address
- Start database (bondora-db-intance)
- Change new IP address of web server to Record Set (www.reporting.studiograni.net) in Route 53

For SSH (if necessary) remember to use new public DNS of web server.

## 6.2 Changes to Django project

Path to Django project: ~/consumer_lending/bondora_invest

After changes to Django project files, restart uwsgi and nginx for the changes to take effect:

```
sudo service uwsgi restart
```

```
sudo systemctl restart nginx
```