Coltpay AWS Architecture

The following AWS services are being used:

1. Route 53 - handles DNS Routing
2. Certificate Manager - handles SSL
3. Elastic Beanstalk - Manages api Application / load balancer and restarts or creates new instances for any downed servers.
4. Application: colt-api
    a. Prod
        i. Api.coltpay.com
        ii. Load balanced - Currently only using 1 server.
    b. Development
        i. dev.api.coltpay.com / dev.coltpay.com
        ii. Load balanced - Currently only using 1 server.
5. Database
    a. RDS - Aurora Database
        i. Colt-cluster-instance-1.cxrqlvyo5nm0.us-east-1.rds.amazonaws.com
            1. Login info:
                a. Colt master database: admin > d9mQNCU5CFXu9yvz
                b. Colt prod database: prod > 2w5XGmwjUtKMAatC
                c. Colt old dev database: dev > 9QxKHub2Vet3WpHZ - no longer used
            2. 35-day recovery
            3. Accessible:
                a. Colt-api > Prod
    b. RDS - Mysql Database new Development
        i. Colt-api > Development
        ii. Accessible:
            1. Colt-api > Dev
6. S3
    a. Coltpay.com / dev.coltpay.com
        i. Private bucket
        ii. /dist folder holds the angular frontend
    b. admin.coltpay.com / admin.dev.coltpay.com
        i. Private bucket
        ii. /dist folder holds the angular frontend
7. Cloudfront
    a. Coltpay.com
    b. Dev.coltpay.com
    c. Admin.coltpay.com
    d. admin.dev.coltpay.com

Notes:

1. All services are in us-east-1
2. System Architecture as follow
    a. Production
        i. Coltpay.com: Route 53 (dns) -> cloudfront (SSL Termination) -> s3 bucket
        ii. Admin.coltpay.com: Route 53 (dns) -> cloudfront (SSL Termination) -> s3 bucket
        iii. api.coltpay.com: Route 53 (dns) -> elastic beanstalk -> load balancer -> ec2 instance
    b. Development
        i. dev.coltpay.com: Route 53 (dns) -> cloudfront (SSL Termination) -> s3 bucket
        ii. admin.dev.coltpay.com: Route 53 (dns) -> cloudfront (SSL Termination) -> s3 bucket
        iii. api.dev.coltpay.com: Route 53 (dns) -> elastic beanstalk -> load balancer -> ec2 instance
3. In this setup, the front end is served from CloudFront via s3
4. The angular front end/admin panel has to be manually uploaded to S3 via drag and drop after the application is built in dev/prod model
5. database server holds production database
6. Separate RDS that holds development database
    a. Admin credentials should never be used to prevent accidental data loss
7. Services are tagged by the “env” key for cost comparisons

○ All “dev” services can be terminated to save on the monthly bill

1. Github projects setup
    a. Colt-frontend - has coltpay website and user backend UI
    b. Colt-admin - has coltpay admin UI
    c. Colt-api - backend web application. Handles cryptoapis integration. Has cronjobs to check for transactions.
    d. Colt-wallet - current not used has old blockchain api integration.
    e. Colt-wp - holds the coltpay wordpress plugin