Python Flask 直播项目实战

Cloud.ye

目　录

- 项目状况介绍
- Flask介绍
- 整体结构
- Flask开发和部署
- 使用的插件
- 第三方接口
- 性能测试
- 小经验

项目状况介绍

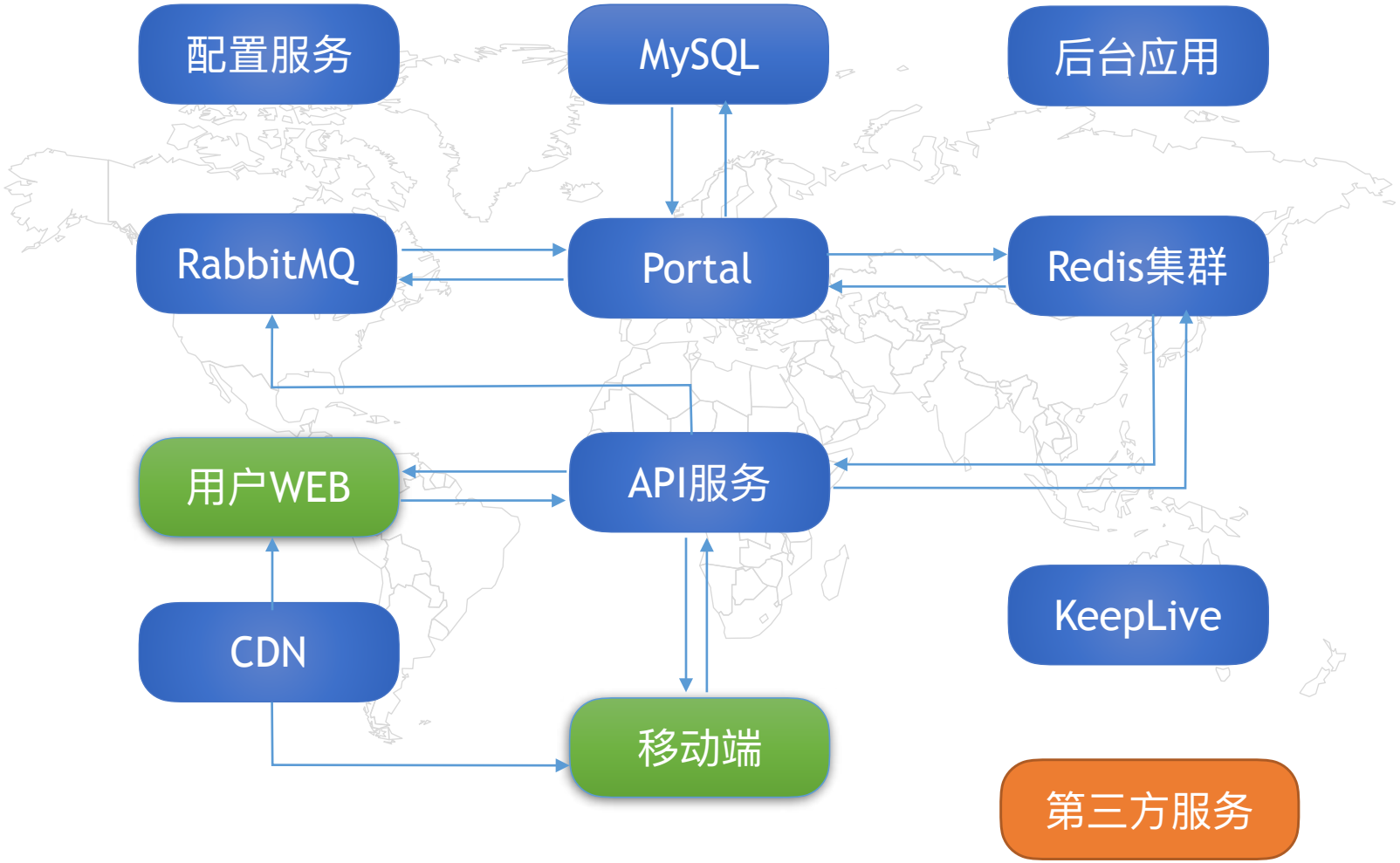
- 狮吼直播，是一款游戏竞技直播软件
- 以游戏赛事、竞技、才艺为主的直播软件

- 进度：项目组刚成立
- 人员：1名UI设计师、1名安卓程序员、1名服务端技
- 时间：要求2周左右出一个DEMO，能播放视频和展示测试数据
- 人员计划：2名UI、2名产品、3名服务端、3名安卓、2名苹果、1名视频编解码、1名项目管理

•Flask介绍

- 轻量级、微框架
- 100% WSGI 1.0 兼容
- 易扩展
- RESTful request dispatching
- 中英文资料详细

整体结构


配置服务
MySQL
后台应用
RabbitMQ
Portal
Redis集群
用户WEB
API服务
KeepLive
CDN
移动端
第三方服务

• Flask开发和部署——组件

用到的组件：

Flask-Limiter	https://flask-limiter.readthedocs.io/en/stable/
Flask-Login	https://flask-login.readthedocs.io/en/latest/
Flask-Mail	http://pythonhosted.org/Flask-Mail/
Flask-Migrate	http://flask-migrate.readthedocs.io/en/latest/
Flask-Redis	
Flask-SQLAlchemy	http://flask-sqlalchemy.pocoo.org/2.1/
Flask-Script	http://flask-script.readthedocs.io/en/latest/
Flask-Bootstrap	http://pythonhosted.org/Flask-Bootstrap/
Flask-HTTPAuth	http://flask-httpauth.readthedocs.io/en/latest/
Flask-WTF	https://flask-wtf.readthedocs.io/en/latest/
Flask-Admin	https://github.com/flask-admin/flask-admin/
celery	http://www.celeryproject.org/
Flask-Moment	
flask-swagger	
itsdangerous	

Flask开发和部署——部署方案

服务器：
centos6.5 or later

阿里云

以前使用：
nginx/1.8.0
uwsgi
python3.5/virtualenv3.5

现在使用：
nginx/1.8.0
gevent (1.1.2)
gunicorn (19.6.0)
python3.5/virtualenv3.5

Flask使用的插件——重点插件介绍

- Flask-Login
  - 用于管理用户的登录/验证/session
  - User Model 通常需要实现几个方法：is_authenticated、is_active、is_anonymous，请参考 flask_login.UserMixin模块，或是 class **User**(UserMixin, db.Model) 继承下来
- Flask-SQLAlchemy
  - ORM很重要，但是不要滥用
  - 为你的Model 留个to_json
  - db.relationship:dynamic和select
- Flask-WTF
  - 不要过度依赖 wtf.quick_from
  - 启用CSRF
- flask-swagger
  - 为API创建文档和客户端库
- Flask-HTTPAuth
  - API服务的 Token验证
- Flask-Bootstrap
  - 为了偷懒和漂亮
- celery
  - 分布式队列，高可用、快速、灵活

第三方接口

视频：金山云

- 视频流管理
- 视频转码
- 鉴权和防盗链
- 数据监测和统计
- 视频截图管理

聊天：融云

- 自定义聊天消息
- 聊天记录监测（黑白名单）

推送：极光

- 角标消息

短信：思空

- 注册短信、活动短信

CDN：网宿

- 金山云的补充

Flask的Blueprint和app_create

蓝图的概念：（引自官方文档）
记录注册到一个应用时的操作执行情况。
当从一个端点到另一端分发请求和生成 URL 时，Flask 关联视图函数和蓝图。

蓝图的作用：
分拆应用，按模块实例化应用功能

```
def configure_blueprints(app):
    app.register_blueprint(main, url_prefix=app.config["MAIN_URL_PREFIX"])
    app.register_blueprint(api, url_prefix=app.config["API_URL_PREFIX"])
    app.register_blueprint(auth, url_prefix=app.config["AUTH_URL_PREFIX"])
    app.register_blueprint(
        manager, url_prefix=app.config["ADMIN_URL_PREFIX"]
    )
    app.register_blueprint(
        message, url_prefix=app.config["MESSAGE_URL_PREFIX"]
    )
……
```

小经验——create_app

```
def create_app(config=None):
   app = Flask(“stream")

   app.config.from_object('stream.configs.default.DefaultConfig')
    # Update the config
    app.config.from_object(config)
    app.config.from_envvar("STREAM_SETTINGS", silent=True)

    configure_celery_app(app, celery)
    configure_blueprints(app)
    configure_extensions(app)
    configure_template_filters(app)
    configure_context_processors(app)
    configure_before_handlers(app)
    configure_errorhandlers(app)
    configure_logging(app)

    return app

……
```

```
def configure_extensions(app):
    csrf.init_app(app)
   db.init_app(app)
    migrate.init_app(app, db)
    mail.init_app(app)
    cache.init_app(app)
    debugtoolbar.init_app(app)
    redis_store.init_app(app)
    limiter.init_app(app)
    ……
```

小经验——扩展管理

extensions.py

```
from flask_limiter import Limiter
from celery import Celery
from flask_login import LogingManager
…

allows = Allows(throws=AuthorizationRequired)
db = SQLAlchemy()
login_manager = LoginManager()
mail = Mail()
cache = Cache()
redis_store = Redis()
debugtoolbar = DebugToolbarExtension()
migrate = Migrate()
plugin_manager = PluginManager()
csrf = CsrfProtect()
limiter = Limiter(auto_check=False, key_func=get_remote_address)
celery = Celery(“stream")
….
```

小经验——找资料

1、少百度、多google/bing
2、多看官方文档
3、多看库的源代码实现
4、找个好的前端伙伴，大大提高效率
5、不要重复造轮子、学习使用好现有轮子
6、写好注释，从代码中生成文档（Sphinx、sphinx-apidoc）
7、在使用1个组件之前，需要先看看组件在GitHub上是否有人维护
8、尽量使用环境变量定义配置

QQ :1472621
WX :1472621
email:1472621@qq.com