**前端架构、认证和应用程序示意图**

* **画一个示意图**，说明应用程序的前端、中间层和后端的设计。

A screenshot of a computer

Description automatically generated

**前端架构**

* 描述前端的组件架构。

**前端是单页Web应用（single page web application，SPA）， 是加载单个HTML 页面并在用户与应用程序交互时动态更新该页面的Web应用程序。用户操作体验好，用户不用刷新页面，整个交互过程都是通过接口请求来操作。**

**主要涉及组件：**

**ReactJS 页面业务逻辑构建**

**Bootstrap： 页面布局**

**AJAX： 前端页面和后端数据交互**

**Cookie： 页面与服务器信任机制，用户登录登出记录用户活动信息**

**Auth0: Auth0 提供 Oauth2 SPA SDK**

* 解释你的用户界面（UI）设计决策。

**页面利用React进行组件化设计和封装，提高复用行和可维护性。 React路由加载各个组件及页面跳转拦截。React DOM进行页面样式布局， Bootstrap 负责响应式布局。利用cookie和session进行权限隔离， 登录非登录控制不同的显示模块。**

**数据库**

* 解释应用程序中哪些数据是持久化并存储在数据库中的？

**用户邮箱，密码，姓名，已经账户余额**

**操作轨迹和记录**

* 如何在应用程序中使用数据抽象层（DAL）的？

**针对MONGO数据库的操作（增删查改）都封装到DAL。 隔离业务逻辑与数据存储。 方便后期替换其他数据库。**

**API**

* 你使用的是RESTful API还是GraphQL API？

**RESTful API**

* 描述你的API端点并解释每个端点的功能。
  + /logout
    - Auth0 logout redirect
  + /callback
    - Auth0 Login redirect callback.
  + /api/create
    - Create Account
  + /api/login
    - Login in with user name and password
  + /api/profile
    - Get user profile
  + /api/update
    - Update user name and balance
  + /api/balance
    - Get account balance
  + /api/depoist
    - deposit the balance
  + /api/withDraw
    - withDraw the balance
  + /api/all
    - Get All user accounts including user nane, email password and balance
* 展示你的API文档。

**部署**

* 你是如何部署应用程序的？
* 应用程序托管在哪里？
* 你是否使用了Docker？