**ChainStack-Platform** Resource Management

Design Document

Version 1.0

**Document Summary**

Author : Mohsin Sayyed

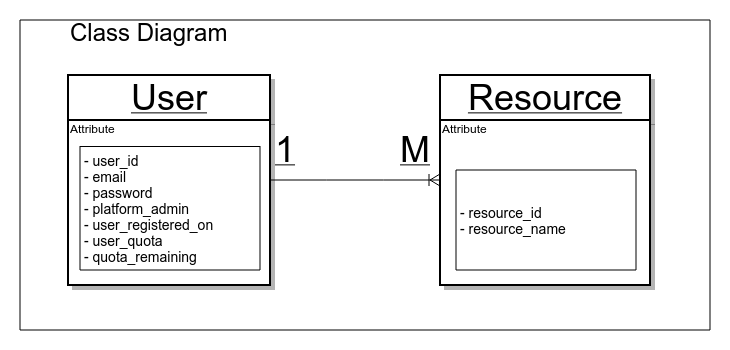
Project Reference: ChainSatck-Platform Resource Management

Document Reference: Interview Take away Assignment

**Revision History**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Description** |
| V1.0 | 18th May 2018 | Final Release |

1. **Platform Design**

**- Technical Choises**

I have used Python Flask micro framework for this assignment

App Design choice

* Flask has the flexibility to use what is required from the MVC  
  architechture and is modularized for designing component based app.
* I have designed this app using Model-Service-Controller pattern
* I have used Blueprint a flask feature which allows for easy app extention by registering blueprint on this app
* For app routing and REST service i have used Flask-restplus registered as a blueprint with this app
* Falsk-SQLAlchemy a file based database is used for record storage and model persistance
* I have used decorator pattern to implement user authentication and authorization to provide restricted access to different users on this platform.
* API include swagger documentation on server which directs in detail for usage

**- Usage Steps**

* Every API on this platform requires to send auth token which the server uses to identify current user and provide restricted access
* Step – 1:

- Login first with user credentials, it will then generate ***auth token***

***-*** use this token for successive api request

* Step – 2:

- Copy auth token and paste in ***\*Authorization*** field for every other api request

- For post request include the appropriate post body from ***\*Model*** definition included in the swagger definition

* Step – 3:

- verify the respone

**2. API Design**

|  |  |
| --- | --- |
| /auth/login | User Authentication Login API  API Type - POST  **Post data -**  {"email":<some-user-email>,"password": <some-user-password>}  **Response Data -**  {  "status": <pass/fail>,  “message”: “respone-message”  “Authorization”: “auth-token”  }  **Status Handled**  200 - success 404 – User Not Found  400 – Bad Request |
| /auth/logout | User Logout API it invalidates the used token  API Type - POST  **Header data -**  “Authorization”: <auth-token>  **Response data -**  {  "status": <response status>,  “message”: <response-message>  }  **Status Handled**  200 - success  403 – Forbidden for Invalid auth token  400 – Bad Request |
| /users/ | GET for List All Platform Users  POST to create new platform users  **API Type** – GET, POST  **Header Data –** Authorization  **Restriction – \*** Only Platform Admin allowed  **Response data -**  **GET response**  {  "data": {  “status”: <response status>  “message”: <response-message>  “data”: [<list-of-all-platform-users>]  }  }  **POST Respose**  {  “status”: <response status>  **“message”: <response-message>**  **}**  **Status Handled**  200 - success  401 – Not Authorized if not admin  400 – Bad request |
| / users / {user\_id} | Single User Related Operations  GET - particular user info  DELETE – delete user from platform  PUT – set user quota  **API Type – GET,** PUT, DELETE  **Query Parameter – user data**  **Header ata – Authorization**  **GET Response** -  **{**  {  **"status": "string",**  **"message": "string",**  **"data": [**  **{**  **"user\_id": <some-id>,**  **"email": <some-email>,**  **"user\_registered\_on": <some-date>,**  **"user\_quota": <some-num>,**  **"quota\_remaining": <some-num>**  **}**  **]**  **}**  **PUT, DELETE Response** -  **{**  {  **"status": <response-status>,**  **"message": <response-message>**  **}**  **Status Handled** -  200 - success  401 – Un Authorized to access other user  404 – User Not Found  400 – Bad Request |
| / resources / | List all platform resources  **API Type** – GET  **Header Data –** Authorization  **Restriction – \*** Only Platform Admin allowed  **Response data -**  **GET response**  {  "data": {  “status”: <response status>  “message”: <response-message>  “data”: [<list-of-all-platform-resources>]  }  } |
| / resources / {user\_id } | User Resources related operations  POST – create a new user resource  GET – Fetch the details of all the resources of  a user  DELETE – delete a particular user resource or all its rresources  **API Type – GET,** POST, DELETE  **Header Data –** Authorization  **Restriction –**    \* A user can access only his info not other  \* A user can create resources only for him    \* A user cannot create resource if quota is  over  \* Only platform admin can set quota  \* Only admin can delete users  \* Platform admin cannot delete his own  account  \* At any given time atleast admin will be there  **Response data -**  **GET response**  **[**  **{**  **"status": <response-status>,**  **"message": <response-message>,**  **"data": [**  **{**  **"user\_id": <some-id>,**  **"resource\_id": <some-id>,**  **"resource\_name": <some-name>**  **}**  **]**  **}**  **]**  **POST, DELETE response**  {  **"status": <response-status>,**  **"message": <response-message>**  **}**  **Status Handled** -  200 - success  401 – Un Authorized for other users  404 – User Not Found  422 – Unprocess entity if user quota over  **400 – Bad Request** |