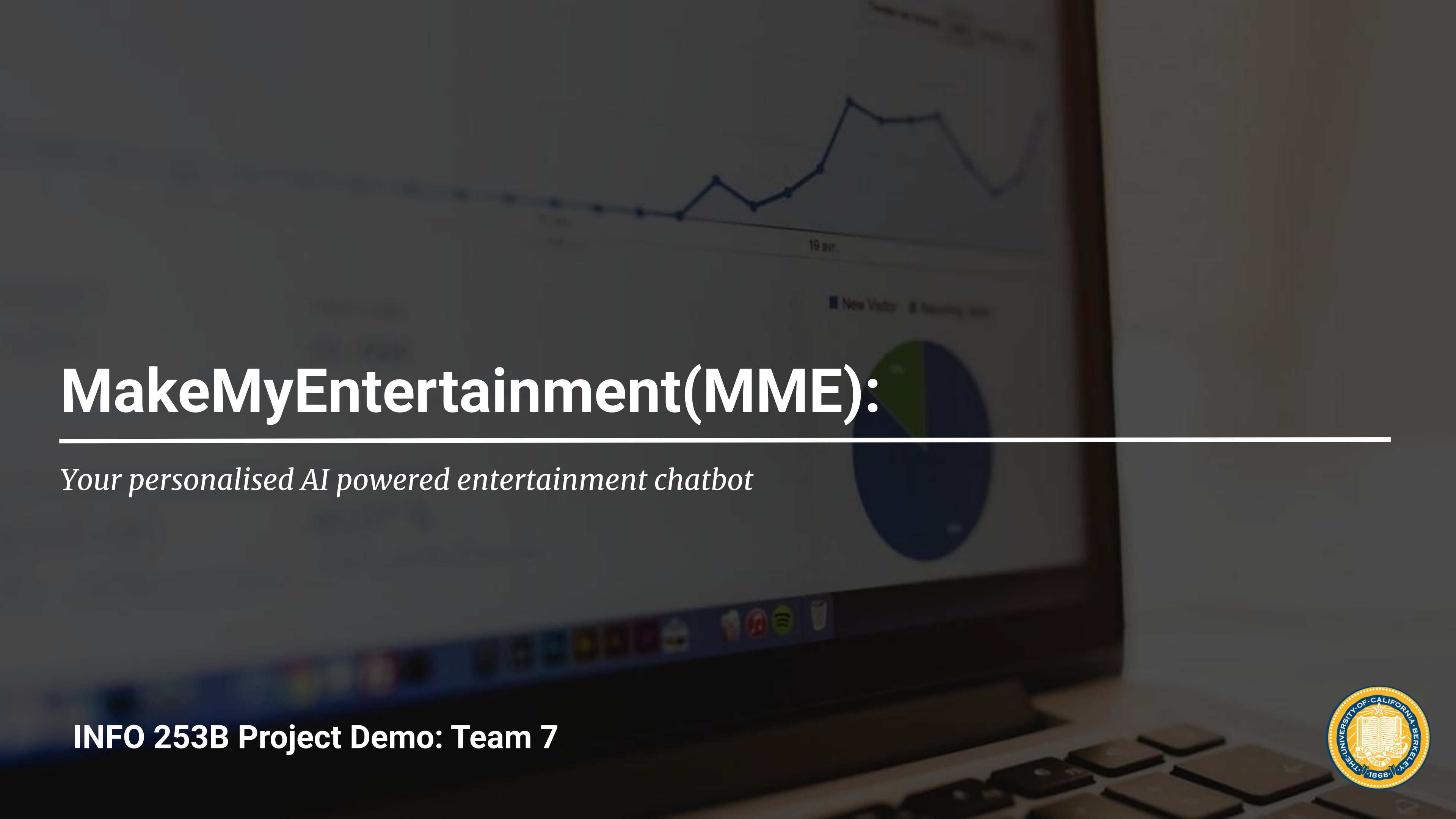


# MakeMyEntertainment(MME):

*Your personalised AI powered entertainment chatbot*

INFO 253B Project Demo: Team 7

# Product Highlights

*For **Customers** of MME:*

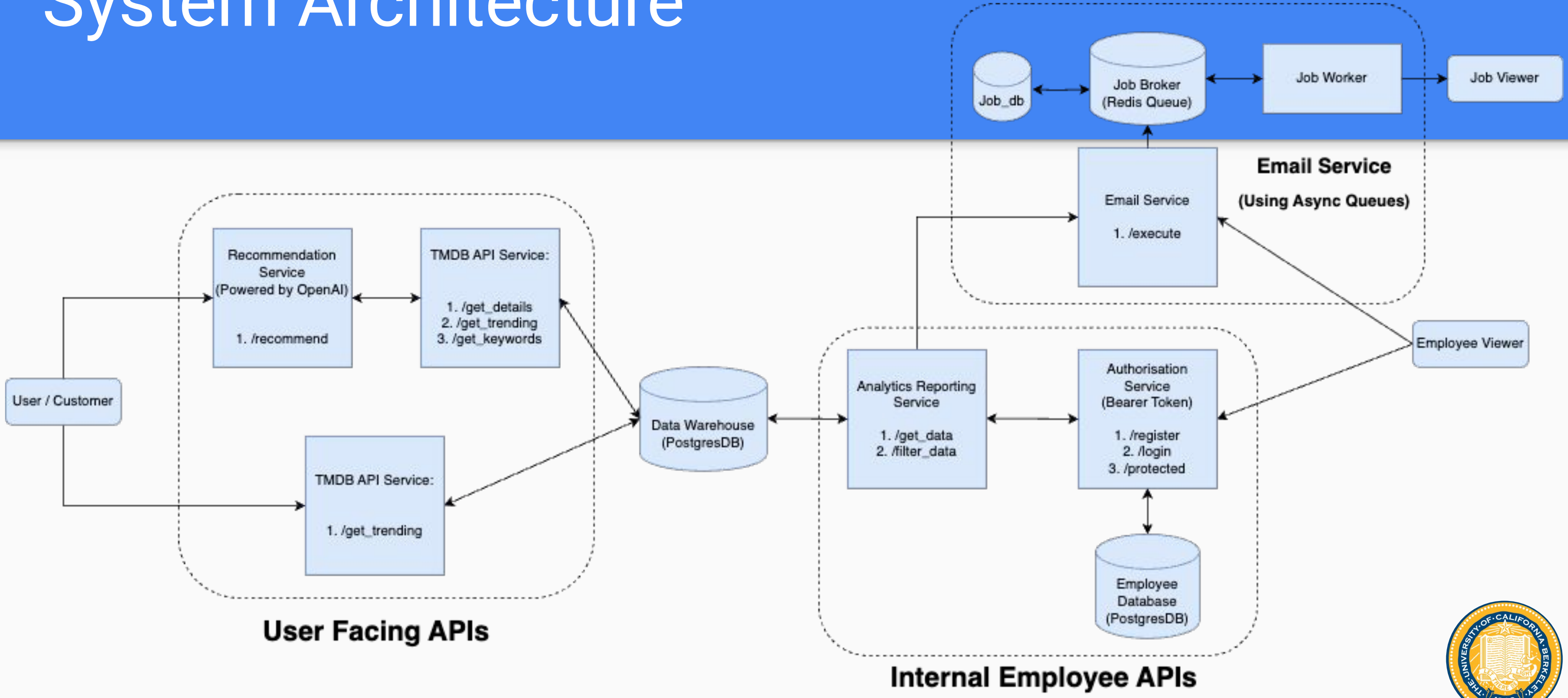
1. Get **trending** movies or TV shows by day or weekly charts.
2. Get personalized recommendations on mood, actors, genre, or whatever you may think of *(powered by OpenAI)*.

*For **Employees** of MME:*

1. Analyse what your users are searching.
2. What movie or TV shows are searched daily.
3. Email daily reports of your service to your Boss.

## System Design overview

# System Architecture



## User Facing APIs

### 1. *Recommendation Service:*

***Endpoint:*** *a.* ***/recommend***

**Purpose:** Uses OpenAI API to recommend tv or movie shows based on some keywords entered by the user.
**Input:** 1. 'preference': Takes mood/keyword/actor/genre. Any valid theme is fine. 2. 'Movie_tv_shows': Movie or TV show.
**Output:** A JSON list of recommendations as generated by OpenAI recommendation engine.

### 2. *TheMoviesDatabase(TMDB) Service:*

**Endpoints:**

**a. /get_details**
**Purpose:** Fetch details of movies/shows as per name and year from TMDB API service.
**Input:** 1. "name": Keyword to search. 2. "media_type": Movie or TV Show. 3. "year": Information of year.\
**Output:** A JSON with details of movie, cast, popularity, release date, etc. Refer documentation for more details.

**b. /get_trending**
**Purpose:** Fetch details of movies/shows which are trending as captured from TMDB API service.
**Input:** 1. "Time_window": day or week. 2. "media_type": Movie, TV Show, People, All.
**Output:** A JSON with details of movie, cast, popularity, release date, etc. Refer documentation for more details.

# Internal Employee Facing APIs

## 1. *Authorisation Service: Token-Based Authentication*

***Endpoints:*** *a.* ***/register b. /login c. /protected***

***Purpose:*** Implements token based authorisation for secure entry into database capturing user activity.
***Input:*** Register your name, email and password. Use mail and password to login.
***Output:*** Generates a token for authorised access into the system after login used for protected endpoint.

## 2. *Analytics Service:*

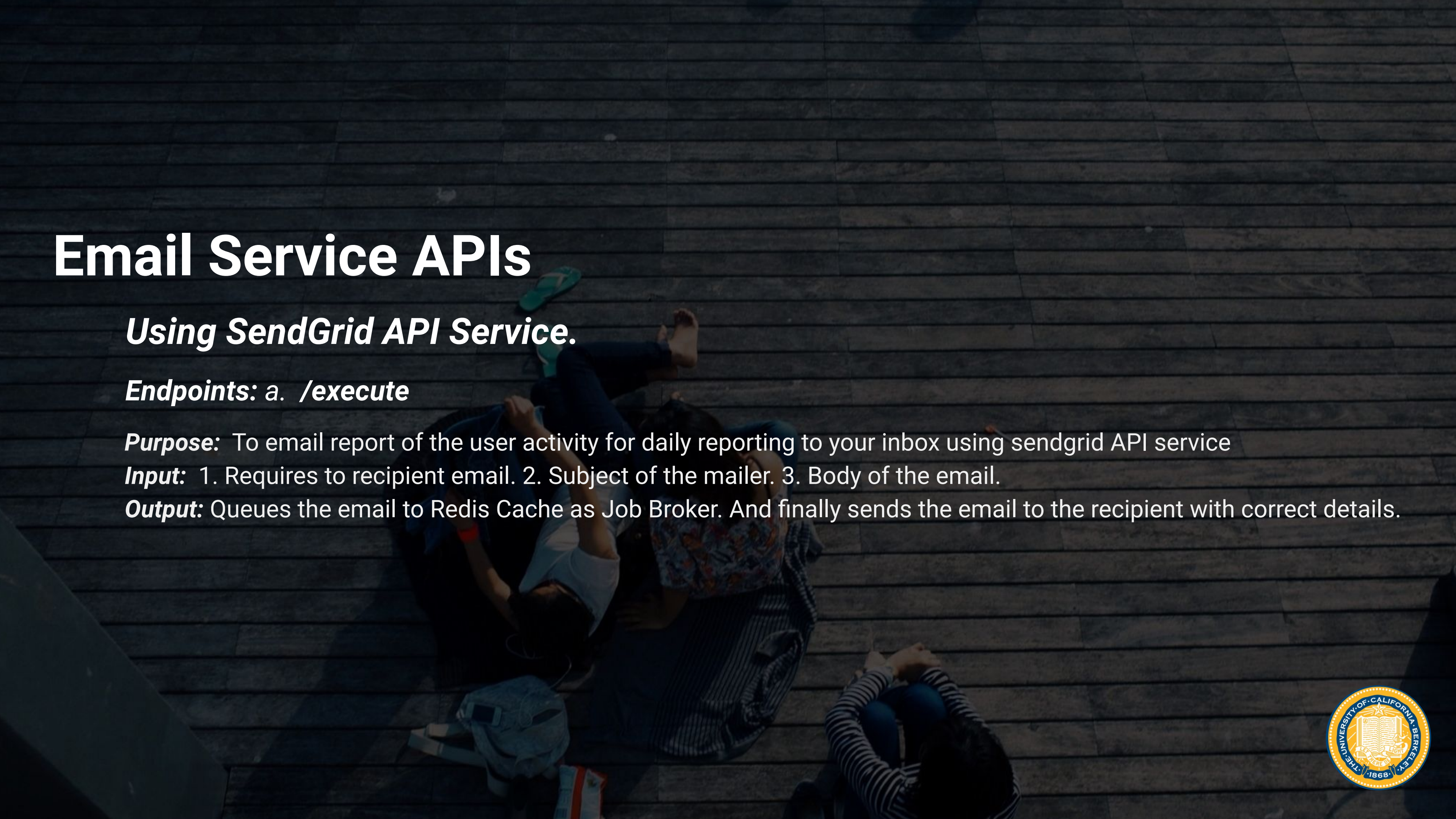
**Endpoints:**

### a. /get_data

**Purpose:** Fetch details of the whole database as is as a report from the persisting postgresDB.
**Input:** Just call the API.
**Output:** A JSON with the whole database as is. As a class project, the size would not be large. Can increase complexity.

### b. /filter_data

**Purpose:** Fetch details of movies/shows based on name, year or genre from the persisting postgresDB.
**Input:** 1. "Time_window": day or week. 2. "media_type": Movie, TV Show, People, All.
**Output:** A JSON with the filtered snapshot of the database as per filters.Refer documentation for more details.



## Email Service APIs

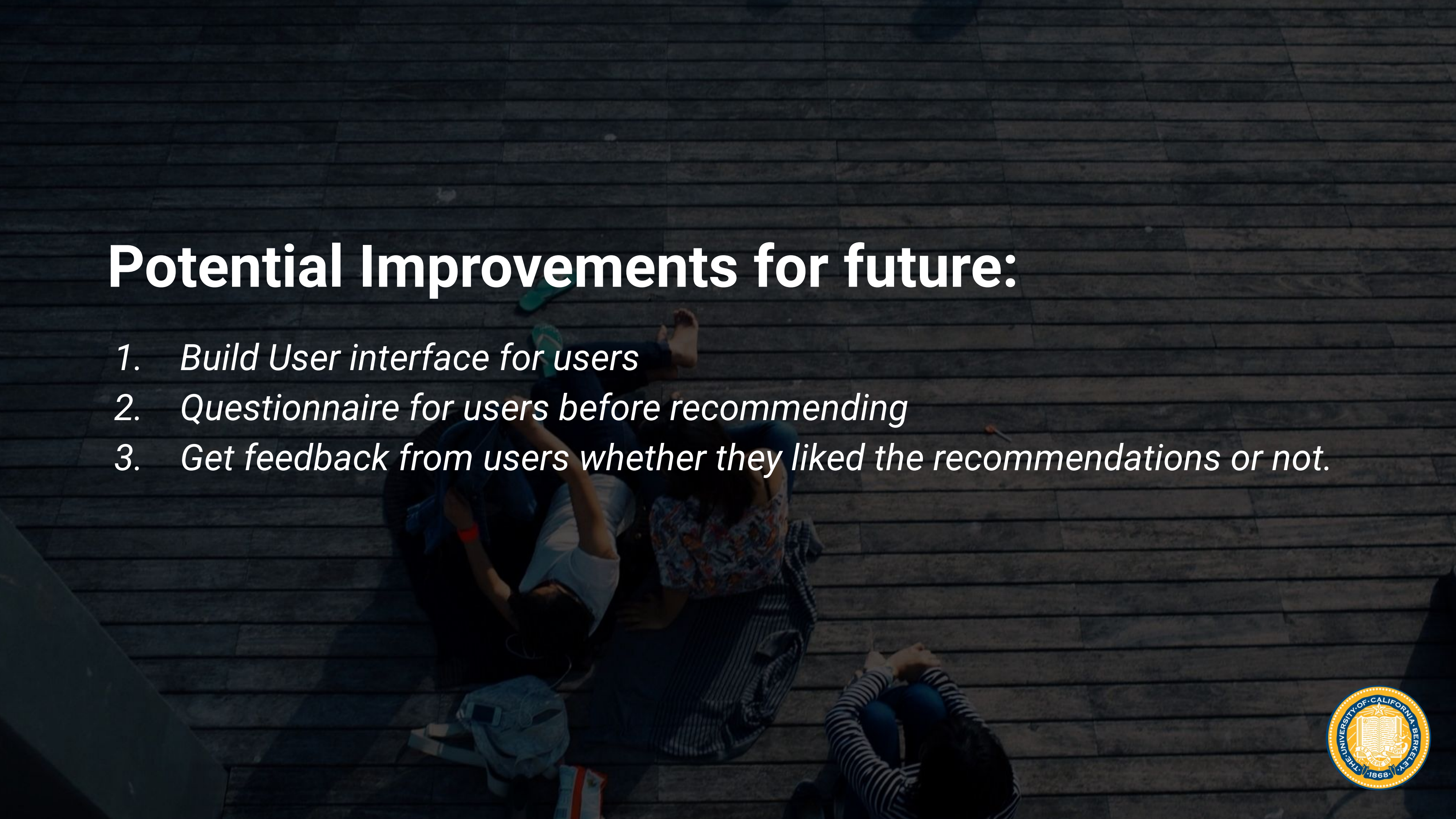
***Using SendGrid API Service.***

***Endpoints:*** *a.* ***/execute***

***Purpose:*** To email report of the user activity for daily reporting to your inbox using sendgrid API service
***Input:*** 1. Requires to recipient email. 2. Subject of the mailer. 3. Body of the email.
***Output:*** Queues the email to Redis Cache as Job Broker. And finally sends the email to the recipient with correct details.

## Implemented Class Concepts & Technologies:

1. *API Development - Flask, Python.*
2. *Authorisation - Token Based Authorisation.*
3. *Async Task Queues - using Redis cache and Celery.*
4. *Storage Systems/Database - Using PostgreSQL and SQLAlchemy.*
5. *Containerisation - using Docker.*
6. *Communicating multiple APIs & Terminal based Chatbot.*



## Potential Improvements for future:

1. *Build User interface for users*
2. *Questionnaire for users before recommending*
3. *Get feedback from users whether they liked the recommendations or not.*

# Team

*Answer the question, "Why are we the ones to solve the problem we identified?"*

Chirag Manghani

Mayank Sethi

German Perea

Yusuf

Prateek Aher