Project Summary

1. Personalized Event Recommendation Web Application

Designed and implemented a web application that to provide event recommendation based on user's favorite history and user's current location.

● Built an interactive web page**(HTML/CSS/JavaScript)** as front-end for users to search event, update preference and view the recommended events.

● Built back-end service with **JAVA**. Utilized six **Java servlets** as controller to handle HTTP requests and responses.

● The system gets the location of the user from user's IP address: `http://ipinfo.io/json` or the geo, and do a search based on the location.

● The system has two kinds of user: unregistered user and registered user. The unregister user cannot enter the main system. The system has authentication. The authentication is able to check whether the username and password are correct. The users must send POST request to the back-end service to register and then sign into the system to get all access.

● All events data come from the **TicketMaster API**:
`https://app.ticketmaster.com/discovery/v2/events.json?apikey={key}&geoPoint={location}`
`&keyword={keyword}&radius={miles}`

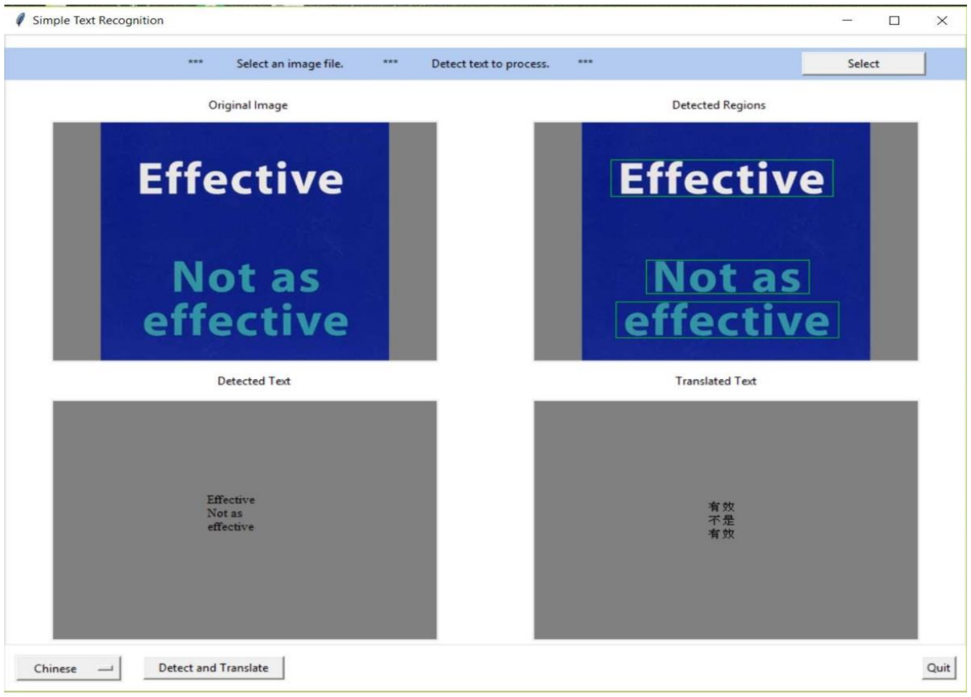
● Created relational database using **MYSQL** to store real time events data fetched from TicketMaster API and user preference information. With **JDBC API**, Java can interact with the **MYSQL** database.

● **The content-based recommender system** uses attributes of the items for a given user has liked in the past to recommend similar items, regardless of the preferences of other users. The system uses content-based recommender algorithm based on the user favorited history. (For example, a user set a sports event, which happen in June, 2019 and it is within 20 mile away from the user's current location, as his favorite. Then the system will recommend sports events which all happened in June 2019, and less than 20 miles away to the user.)

● Create an ec2 instance and deployed the web application on Amazon **AWS**. DEMO:
http://ec2-3-135-236-15.us-east-2.compute.amazonaws.com:8080/BestPlace/
Test user: username: ryan   password: 123456

2. Words Detection and Translation Application

Created a desktop application that to detect English words in an image and to translate them into different languages.



Effective
Not as
effective

有效
不是
有效

● Build a graphical user interface with **Python Tkinter**. The UI contains canvas, frames, label components. The user can input an image with the 'Select' button.

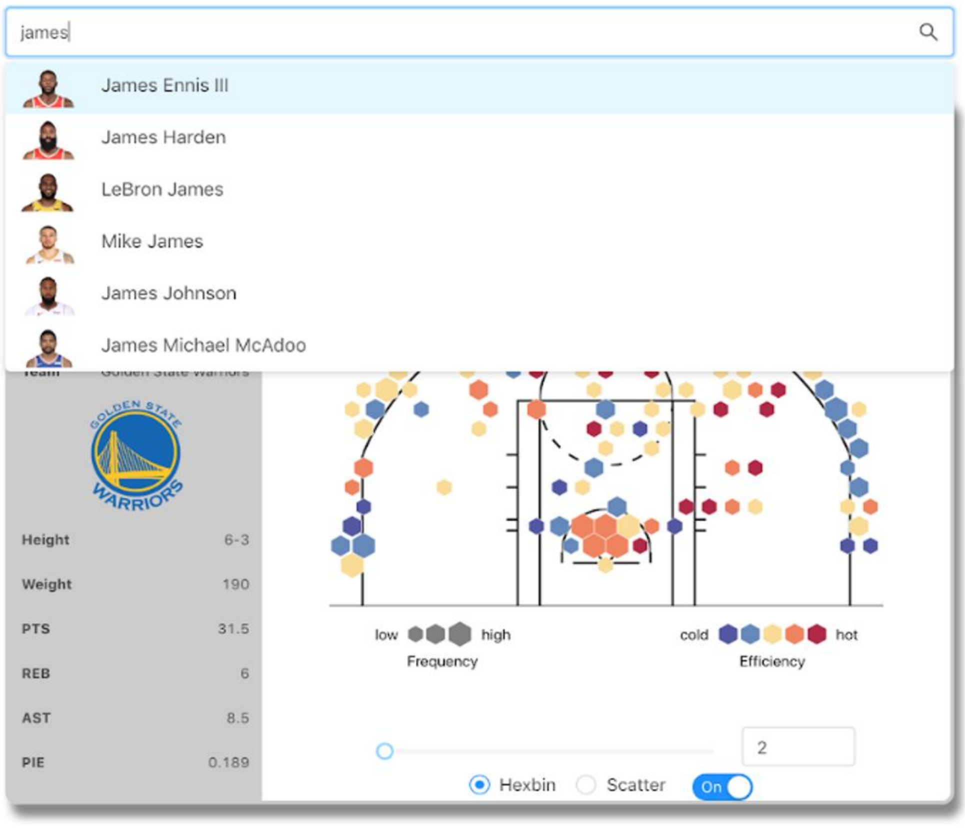
● Utilized **OpenCV** to detect region of texts. The application uses **Digital processing Technique** (Gaussian Blurring, Adaptive Thresholding, Dilation, Flood Fill etc) to get all the contours, then sort the contours.

● Contours sorted by y values, then x. (x, y is the location of the contours in the image). Contours with different ratio or too small/big filtered out. height/width < 0.5 for lines of text is the region of interest. if (((I.shape[1] * 0.98) > h > avgh - (avgh * 0.45)) and (w > avgw - (avgw * 0.65))) and (0.01 < h/w < 0.5)

● Convert the image into string with **Tesseract.**

● Integrated **Google Translate API** to translate all the strings into different languages

3. React JS Based NBA Player Strength Display Application
Built **front-end web application** with **JavaScript** to provide strength visualization of NBA player

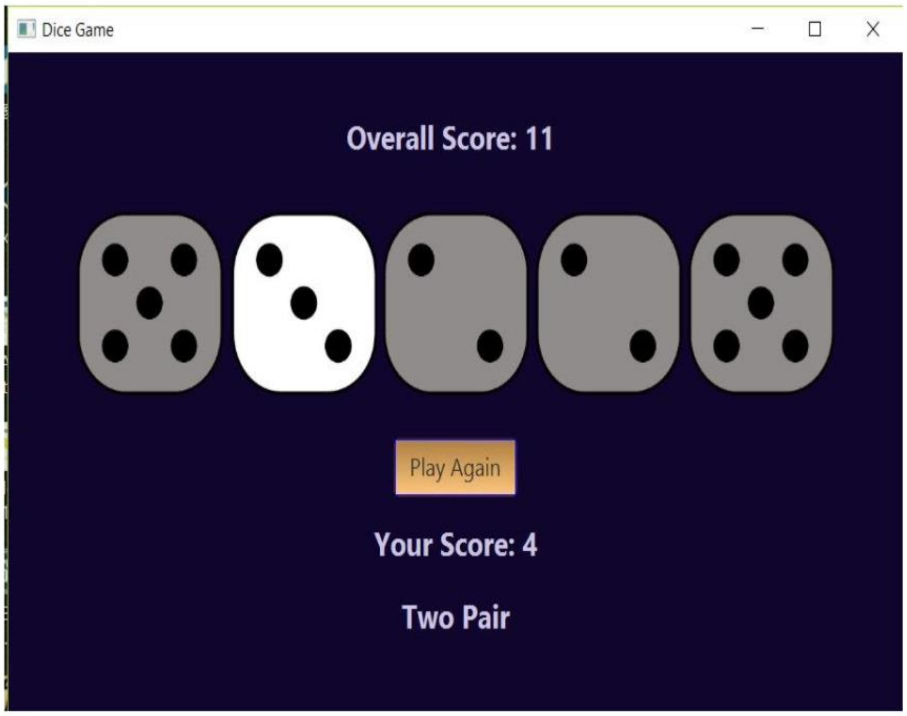


● Create a dashboard using **React, D3**, and **Ant Design**
● Fetch data with **stats.nba.com API** (npm install nba).

```
const NBA = require("nba");
const curry = NBA.findPlayer('Stephen Curry');
console.log(curry);
```

● To visualize individual player's shot data, including a shot chart and profile. I use **d3-shotchart API**(npm install d3-shotchart). This API interact with NBA shot data and become a visualization chart. It helps user identify trends, strengths, weaknesses of the individual player and ultimately improve the player overall shooting percentage.
● Optimized the user experience by adding a autocomplete player search bar (**Ant Design**) providing a list of player(image and name) in the suggestion list.

4. Dice Game
Created a Five-Dice Game desktop application with **JAVA FX and CSS**.



● The application contains 5 Dice Object in the ArrayList.
ArrayList<Dice> diceList=new ArrayList<Dice>(sizeOfArray);
● All Dices are at the initial state 1 at the beginning of Game.
● Each Dice has Mouse_Clicked EventHandler, once mouse clicked, it is held and become gray. It will not roll the next time.
● The player has three rolls per turn, once three rolls are up, the application will display the "best hand" the user has and how many points they received this round. Also will show user overall score they received(overallScore=overallScore+score;).