1.Clear demo presentation. Arriving on time and completing the presentation within the given time frame (details will come at a later time on campuswire). (+1%)(All)

2.A clear demo featuring a user's end-to-end process interacting with the system that involves presenting the CRUD operations, the advanced database program, and the (optional) creative function (10%)（Yuese/Junho)

3.if you implemented stored procedure+trigger (12%):(Jiatong)
Stored procedure requirements (8%):
A complete and functioning stored procedure (2%),
involves at least two advanced queries (2%, 1% each),
uses cursors, involves looping and control (e.g., IF statements) structures (2%),
provides useful functionality to the application (2%).
Trigger requirements (4%):
A complete and functioning trigger (1.5%),
involves event, condition (IF statement),
and action (Update, Insert or Delete) (1.5%),
provides useful functionality to the application (1%).

Extra credit:
4.The entire application is hosted on GCP AND connected to a MySQL database hosted on GCP. (+1% for hosting both application and database on GCP. Hosting both the application and database on other platforms, including but not limited to Azure and AWS, will receive this extra credit upon the project TA's approval.) (Yuqing)

5.A functioning and interesting creative component that is relevant to your application (up to +2% of the entire project grade)(Junho)

During the demo, the team should also discuss the following points: (-1% for any missing point)

1.（Jiatong）Explain your choice for the advanced database program and how it is suitable for your application. For example, if you chose a stored procedure+trigger, explain how this choice is suitable for your application.

A：I choose an advanced database program that help to detect the proper locations to be added to the saved folder for a user. This stored procedure first retrieve all subscribed locations for the current user. It then calculates to check whether the current new location is too close to a saved location. If the location is too close, the saving action will be denied. If the location is not that close, the saving action will happen as normal.

The Trigger I choose is a trigger on the User table, where we first detect whether the email and the password have already been used. If we want to add a new account with an existing email address, then that event will be denied.

The transaction transaction option does not quite fit our program because our program is a user-based interaction program that rely solely on exclusive manipulations that is related to individual user accounts. As a result, the conflicts of read and write will not cause any problem, and adding a transaction control does not extend the functionality of our program.

2.（Junhong）How did the creative element add extra value to your application?

A：The creative element I’ve add is favorite folder. By clicking the button in the rentInfo displayed in each row, the UserID, Tract, Amount and Corresponding Year could be input in Favorites table I’ve created in SQL database. In addition, clicking this button could upload the UserID, LAT, LON to Subscription table in SQL database, accomplished by calling the Storage&Trigger Query “AddSubscription”. This query could avoid uploading data when the distance counted in query is smaller than the preset maxDistance. In other words, if there are any location near the location already in Subscription table, it could stop adding it to favorites.

This function could let users to save the preferred rentInfo for further investigating this place, and could avoid unnecessary redundancy of saving the nearby location in the favorite folder.

3.（Yuqing）How would you want to further improve your application? In terms of database design and system optimization?

A：

Database Design:

1. **Normalization**: Reducing redundancy to prevent anomalies.
2. **Partitioning & Caching**: Utilizing partitioning and caching for large datasets to enhance performance.

System Optimization:
1. **Code Optimization**: Profiling and refactoring code to improve efficiency.
2. **Multithreading**: Allowing simultaneous connections through multithreading.
3. **Load Balancing**: Distributing application load across servers to prevent overloading.
4. **Caching & CDN**: Using caching and Content Delivery Network (CDN) to save processing time and reduce latency.

4.（Yuese)What were the challenges you faced when implementing and designing the application? How was it the same/different from the original design?

A：550 N Figueroa street 90012

Time may be the biggest challenge for our entire team. We need to combine two different sets of data (crime rates and rental information in Los Angeles) in a short period and find an effective and rapid way to meet the project requirements.

As for myself, I also need to design a simple, beautiful, and functional UI in a very limited time. I have utilized some skills I previously learned, such as Photoshop, to assist me in that. This is my first time designing, and I hope this UI can enhance the interactivity of our team's application.

Originally, our application was planning to include some filter features, and rental information would be displayed on the homepage through images. However, due to time constraints, we have made changes with using a favorite folder feature and an interactive map with accompanying text. This new strategy has brought more interactivity to our application and allows users to access information more quickly.

5.(someonehelp！！)If you were to include a NoSQL database, how would you incorporate it into your application?

A：

Incorporating a NoSQL database into your application involves several steps:
1. **Identify Use Cases**: Determine where NoSQL fits, such as unstructured data or real-time processing.
2. **Choose the Right Type**: Select the NoSQL database type (e.g., key-value, document) that aligns with your needs.
3. **Integration**: Establish connections and ensure compatibility with existing systems.

4. **Update Application Logic**: Modify or create logic to interact with NoSQL, using appropriate APIs and query methods.
5. **Security**: Implement relevant security measures like authentication and encryption.
6. **Testing and Optimization**: Test integration to meet performance and reliability requirements, and optimize as needed.
7. **Monitoring**: Utilize tools to monitor performance, manage backups, and updates.

*You must submit your code to the repository in order to receive the grade. You should also tag your release. Failure to tag your release will result in a 3% deduction.
All members must be present at the final project demo. Members who did not attend the entire duration of the demo will receive a 0