5/5/2020 CS1D Project 2 UMLs

mainWindow **CLASS DIAGRAM** *ui : Ui::MainWindow *m controller : Controller *m graph : graph selectedStadiums: QVector<QString> sortedStadiums : QVector<QString> totalDistance : double trip : tripType on_ClearBtn_clicked(): void on_Loginbtn_clicked(): void keyPressEvent(QKeyEvent* pe): void on gotodisplaybtn clicked(): void on teamcombobox currentTextChanged(const QString &arg1): void on displaystadiumteambtn clicked(): void on disstadiumstadiumbtn clicked(): void on disAmericanTeam clicked(): void on disNationalbtn clicked(): void on_disstadiumtypobtn_clicked(): void on disteamopenroof clicked(): void on disstaddatebtn clicked(): void on disstadiumseats clicked(): void on disstadiumlarcenter clicked(): void on disstadiumsmalcenter clicked(): void on loadTeamInfo clicked(): void on_loadSouvenirInfo_clicked(): void on Backtoadmin clicked(): void on loadDatabtn clicked(): void on_loadstadiumsbtn_clicked(): void on backtopass clicked(): void on_AddSouv_clicked(): void on Souvenirbtn clicked(); void on BacktoAdmin clicked(): void on TeamCombosouv activated(const QString &arg1): void on deleteSouvbtn clicked(): void on_Souveniralltable_activated(const QModelIndex &index) : void Admin on Editsouvbtn clicked(); void Graph on_EditMlBbtn_clicked(): void on stadiumcombobox_activated(const QString &arg1): void QString m username; distance: int on Backtoadmin 2 clicked(): void QString m password; control: Controller on_editstadium_clicked(): void int m_rank; vNum: int on DFSbtn clicked(): void matrix: int** on BFSbtn clicked(): void on MSTbtn clicked(); void username() const: QString minKey(int key[], bool mstSet[]): int on_backtologin_clicked(): void setUsername(const QString &username): void printMst(int parent[]):QVector<QString> on custom button clicked(): void sortStadium(stadiums v): QVector<int> password() const: QString on dodgers button clicked(): void setPassword(const QString &password): void DFSUtil(stadiums start, bool visited[], QVector<QString> &dfsPath): void on_add_button_clicked(): void rank() const: int addEdges(): void on delete button clicked(): void setRank(int rank): void getDistance(): int _primMst(): QVector<QString> usernameChanged(): void bfsTraversal(stadiums start): QVector<QString> passwordChanged(): void DFS(stadiums start): QVector<QString> rankChanged(): void

5/5/2020 CS1D Project 2 UMLs

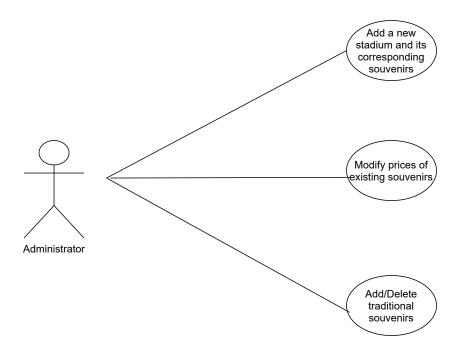
Controller listOfTeams: QList <Team*> listOfSouvenirs: QList <Souvenir*> listOfStadiums: QList <stadiumdis*> m database: QSqlDatabase createTable(): void readTeamFile(): bool readSouvenirFile(): bool readStadiumsFile(): bool loadteam(): void loadStadiums(); void loadSouvenirs(): void updateSouvenir(QString item, QString team, QString price): void deleteSouvenir(QString item,QString team): void createTeam(QString teamname ,QString stadium, QString seatingcapacity,QString location ,QString playingsurface ,QString league, QString date ,QString distancetocenter, QString ballparktypology,QString rooftype): void createSouvenir(QString teamname ,QString item, QString price): void createstadiumdis(QString stadium1,QString stadium2, QString distance): void updateMlb(QString team, QString stadium, QString capacity,QString location, QString surface, QString league, QString date, QString dtc, QString typo, QString roof): void *getstadiumQueryModelsortedbyteam(): QSqlQueryModel *getteamQueryModel(QString team): QSqlQueryModel *getstadiumQueryModelsortedbystadium(): QSglQueryModel *getAmericanLeagueTeams(): QSqlQueryModel *getNationalLeagueTeams(): QSqlQueryModel *getStadiumQueryModelsortedbyTypology(): QSqlQueryModel *getTeamswithopenroofquerymodel(): QSqlQueryModel *getStadiumQueryModelsortedbydate(): QSglQueryModel *getStadiumQueryModelsortedbycapacity(): QSqlQueryModel *getStadiumQueryModelsortedbydistancetocenterLAR(): QSqlQueryModel *getStadiumQueryModelsortedbydistancetocentersma(): QSqlQueryModel *getSouvenirQueryModel(QString team): QSqlQueryModel *getAllSouvenirQueryModel(): QSqlQueryModel *getMlbinfoQueryModel(QString stadium): QSqlQueryModel GetDistBtwnStadium(QString start, QString end): int CheckEdge(QString start, QString end): bool teamName: QString stadiumName: QString seatingCapacity: QString teamLocation: QString playingSurface: QString

league: QString

dateOpened: QString distanceToCenter: QString ballparkTypology: QString teamRoofType: QString

setTeamName(QString name): void getTeamName(): QString setStadiumName(QString stadium): void getStadiumName(QString stadium): void getStadiumName(): QString setSeatingCapacity(): QString setSeatingCapacity(): QString setLocation()CString location): void getLocation(): QString setPlayingSurface(QString surface): void getPlayingSurface(): QString setLeague(QString League): void getLeague(): QString League): void getDateOpened(OString date): void getDateOpened(): QString setDTC(QString dtc): void getDTC(): QString setDTC(QString dtc): void getDTC(): QString setRoofType(QString rooftype): void getRoofType(QString rooftype): void getroofType(): QString

Use Case Diagram



Big O(mega)G's Use Case Diagrams

Case #1

Use Case Name: Add new stadium and its corresponding souvenirs

Use Case Description: A new MLB stadium is added along with its corresponding souvenirs through the administrative system.

Primary Actor: An administrator

Application: This function can only be used by an administrator as it resides within the Administrative menu

Precondition: The current user must be an administrator. This can only be possible by means of the correct login information.

Postcondition: The new stadium(s) will be added to the database along with the respective souvenirs. The admin and the baseball fan will be able to view the new stadium and souvenirs if they press a certain button or buttons to view team information.

Trigger: The "Load Data" button will be clicked. Afterwards, the "Load Stadiums" button will be clicked.

Basic Flow: Once the "Load Stadiums" button is clicked, the admin will get the chance to read the new stadium and its respective souvenirs from an input file. The application will grant the admin to access the file through his or her file system. Once the admin selects on the correct file, then the new information will be stored into the database containing all of the teams' information.

Alternative Flow: Slight issues can arise during this process:

- Admin inputs the wrong credentials. If this is the case, then the admin cannot access the administrative system unless he inputs the correct data.
- QT cannot connect to the database which will impact the tasks that the admin wants to accomplish.
- The file system does not open after clicking "Load Stadiums" due to connection issues or the database file missing.

• The admin does input the new information, but the database does not store the newly processed data.

• The admin does input the new information, but the database already contains the information. If this is the case, then an error message will open or will not allow the admin to re-add the already stored data.

Case #2

Use Case Name: Change prices of traditional souvenirs

Use Case Description: The prices of each stadiums' souvenirs can be modified.

Primary Actor: An administrator

Application: This function can only be used by an administrator as it resides within the Administrative menu

Precondition: The current user must be an administrator. This can only be possible by means of the correct login information.

Postcondition: The prices of traditional souvenirs are edited and saved in the database. The admin will be able to view the new prices. The fans will also be able to view the newly edited prices during their trips.

Trigger: The "Edit Souvenirs" button will be clicked. Afterwards, the souvenirs table will be displayed.

Basic Flow: Once the "Edit Souvenirs" button is clicked, the admin will get the chance to edit the prices of each souvenir corresponding to a particular team/stadium. The admin is able to manually adjust the prices by inputting the proper numerical value.

Alternative Flow: Slight issues can emerge during this process:

• The souvenir table is not displayed due to database connection issues. If this is the case, then the admin will not be able to carry out the desired task.

• The admin does not select a particular team/stadium. This situation will lead to prices not being edited.

• The admin inputs non-numerical data, a negative number, or a number exceeding realistic price values (such as \$1,000,000). An error message prompting the admin to input proper values will appear if any of these conditions are met.

Case #3

Use Case Name: Add and delete traditional souvenirs

Use Case Description: New souvenirs can be added to the database. In addition, old souvenirs can be deleted from the database.

Application: This function can only be used by an administrator as it resides within the Administrative menu.

Precondition: The current user must be an administrator. This can only be possible by means of the correct login information.

Postcondition: The new souvenirs are added to the database. The old souvenirs will be removed from the database. The admin will be able to verify that these tasks have been processed successfully by viewing the souvenir table.

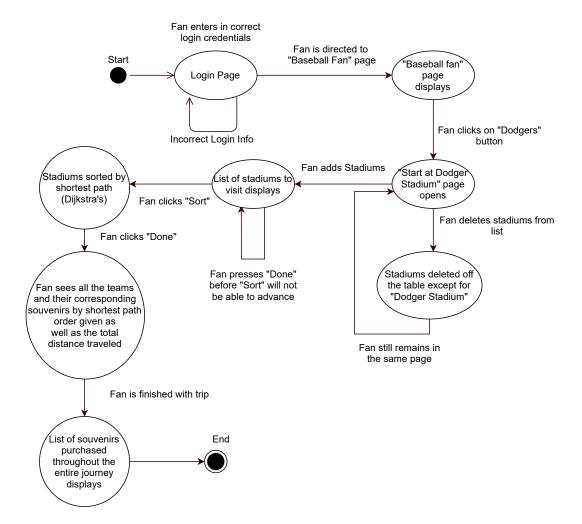
Trigger: In order to add new souvenirs, the "Load Data" button will be clicked first and the "Load Souvenirs" button second. To delete souvenirs, the admin must click the "Edit Souvenirs" button which will result in the souvenirs table being shown.

Basic Flow: Once the "Load Souvenirs" button is clicked, the admin will get the chance to input the new souvenirs by selecting the appropriate file through his/her file system. Once this procedure is met, then the newly added souvenirs will exist in the database. After clicking on "Edit Souvenirs" to delete old souvenirs, the admin has the ability to delete the particular souvenir(s) after selecting the team/stadium that he/she wants to choose from.

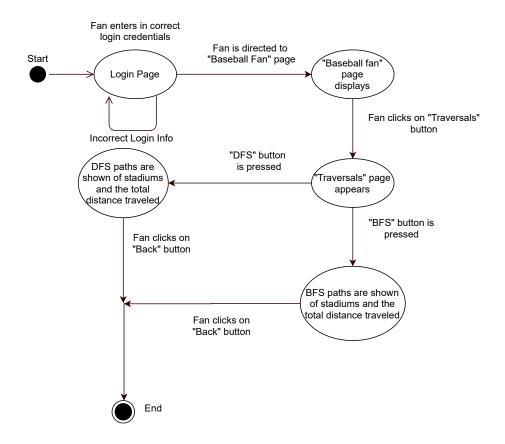
Alternative Flow: Slight issues may occur during this process:

- The souvenir table is not displayed due to database connection issues. If this is the case, then the admin will not be able delete any souvenirs.
- The admin does not select a particular team/stadium. This situation will also lead to the souvenirs not being erased.
- The file system does not appear after clicking "Load Souvenirs" due to database issues. This will result in the admin not being able to add any new souvenirs.
- The admin mistakenly tries to add new souvenirs twice. If this occurs, then the system will give a warning that the souvenirs have already been added.

Fan visits any other team of their choice starting at Dodger Stadium using Dijkstra's for Shortest Path (State Diagram #1)



Fan views the paths taken using DFS and BFS along with their assoicated mileages (State Diagram #2)



Admin is able to modify stadium information such as capacity, stadium name, roof type, etc. (State Diagram #3)

