Brandon Mai
Duc Doan
CS400 Fall 2019
Project Step 7: Portfolio Assignment
URL: http://flip3.engr.oregonstate.edu:8899

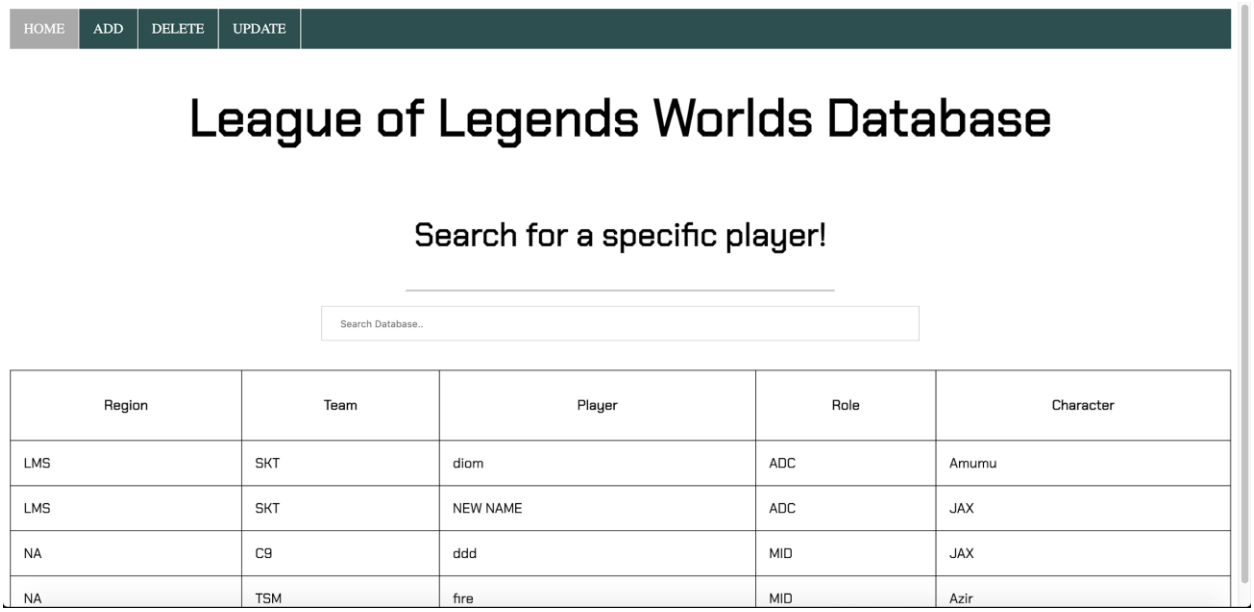
League of Legends Worlds Database

EXECUTIVE SUMMARY

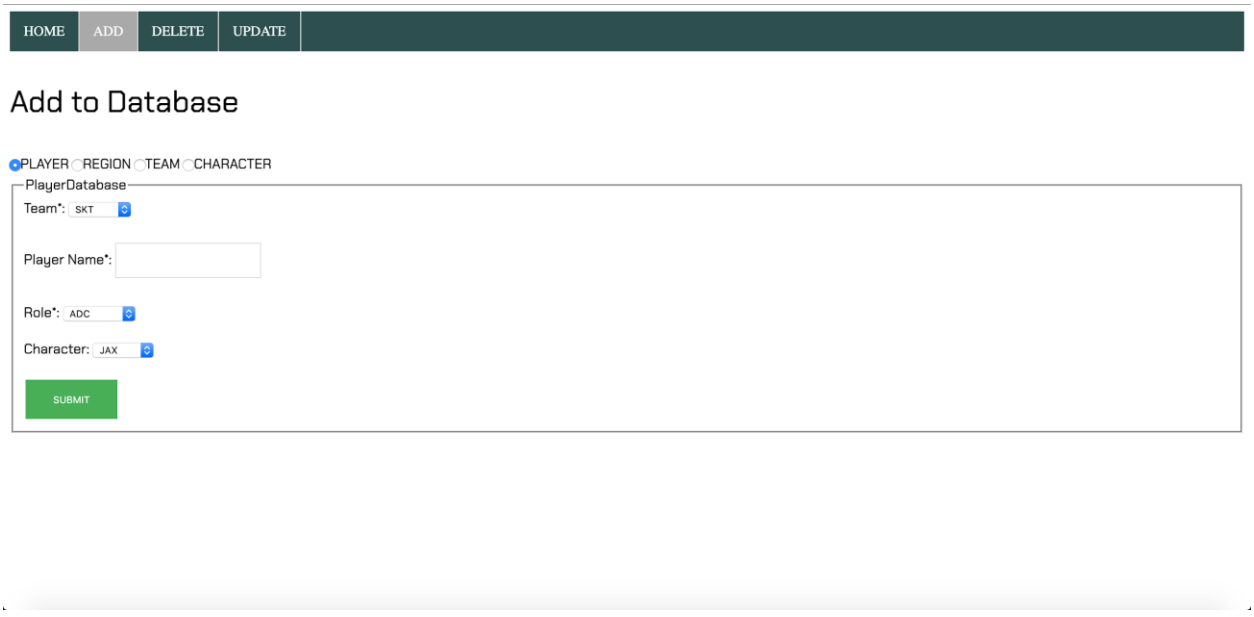
Step 1 to 3	● Did Initial database ○ Corrected major data type variables errors ○ Corrected ER, Schema, and relationship models ○ Removed and added tables and entities (e.g “Regions’)
During our earlier renditions of the project, we had a lot of major details that we failed to consider. From our initial concept, there were data types and redesign we did to best prepare ourselves for our future projects. The Peer Review was extremely helpful in the improvement of our database.	
Step 4	● Fixed and improve Static HTML webpage UI ○ Improve page navigation and adding functionality. ● Fixed and edited Major SQL Create/queries typo
Week 8, we focused on two major aspects, preparing our back end work via the SQL database edits and fixes and improving how we want to design and edit our web page structure. This was one of the major initial step in developing the user experience of our project.	
Step 5 & 6	● Incorporated node.js ○ Updated Navigation/functionality//Web pages to account for Search(home), add, delete, update ● Progress/Improvement in CRUD UI functionality ○ Search/Add finalized and fully implemented. ○ Delete was implemented once we added (“Delete on Cascade”) ○ Major issues with update
We encounter a huge hurdle/block to convert Static HTML to node.js and how to set up the template for this feature. We had issues trying to incorporated SQL into our functionality. We altered our static HTML design to better account for the node.js CRUD functionality. Peer review was a helpful method to find errors and offer advice.	
Step 7	● Finalized CRUD Functionality ○ Search is able to filter through and find player name ■ We have no been able to implement a proper way to show many to many Playes_characters ○ Add is able to add all major entities and tables. ■ Along with foreign intermediate tables ○ Delete is able to delete player and corresponding foreign keys ○ Update is able to update listed variables based on player_id
As for the final wrap up of the project, we implemented many features of CRUD functionality (READ/CREATE/DELETE/EDIT). Due to complications we did focus more on our interface to be based around the player entity. Most of the UI/webpage functionality will allow the user to directly interact based on the player table. CREATE Functionality is the only exception since it allow players to add to all tables. All in all, we are content with the outlook and overall design of the webpage.	

Screenshots of UI Pages

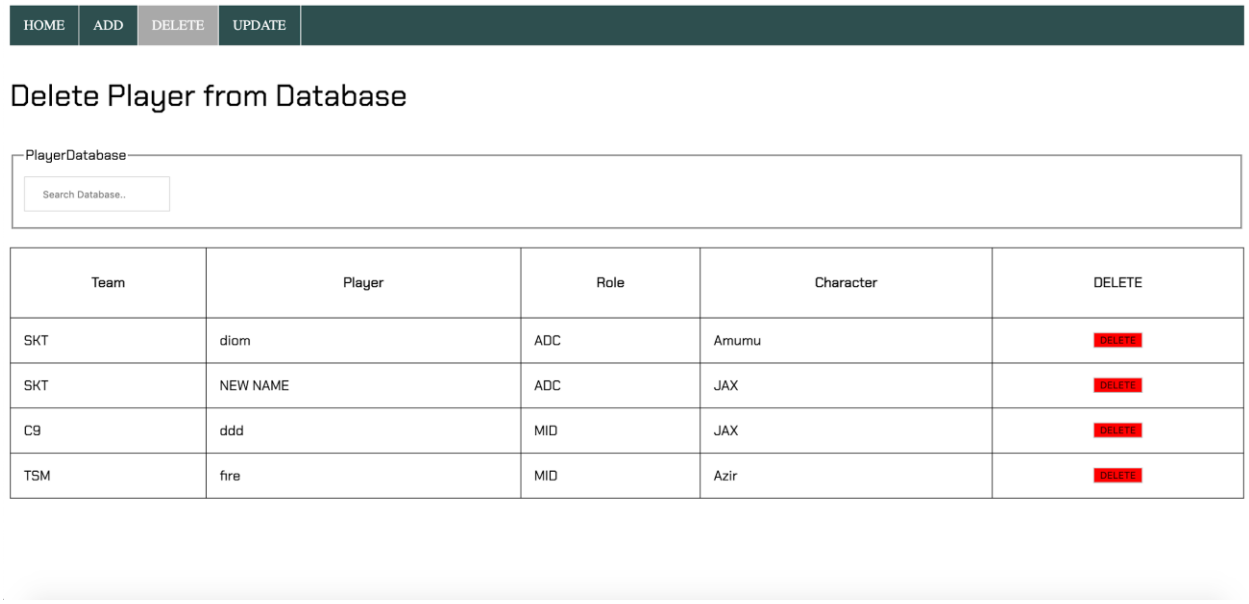
Home Page (Display from CRUD)



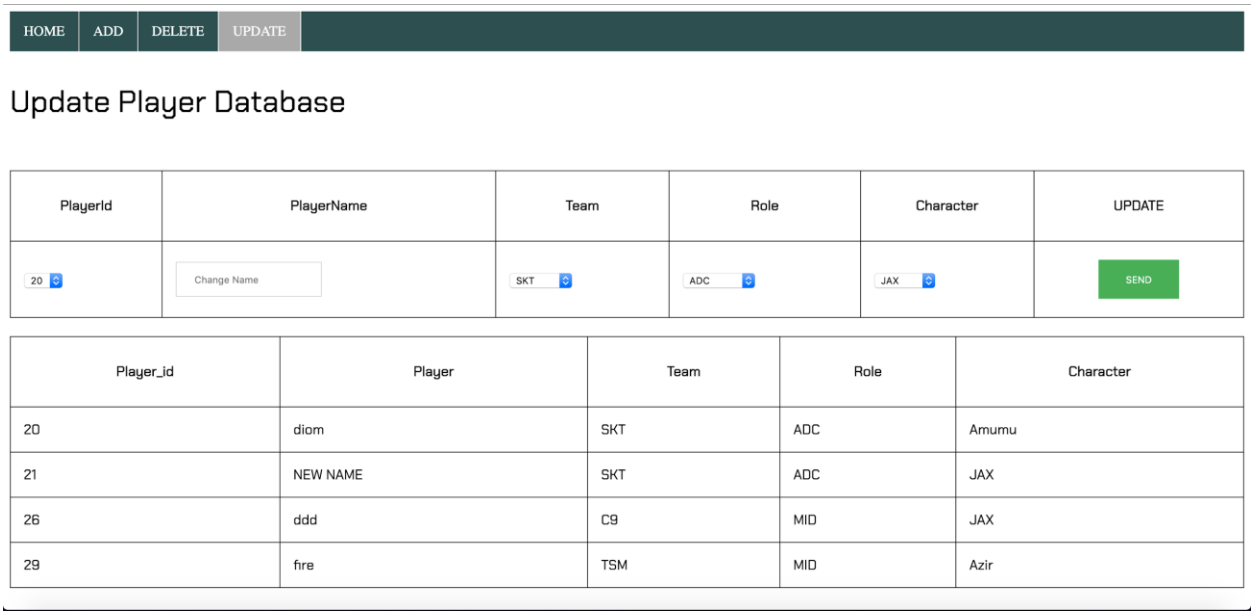
Add Page (Create From CRUD)



Delete Page (Delete From CRUD)



Update Page (Update From CRUD)



Project Outline and Database Outline -

Overview:
Our database will be used to overlook and create a database of all the regions along with teams, players, roles, and characters in respects to League of Legends. The database will be used to highlight the players and teams with their respective roles and characters. The Website will be used to filter and showcase the database. The Website can be used to update and add the following entities (Region, Teams, Players, Characters), and delete entities such as Players.

DataBase Outline (Words):

The database will contain the following entities: Regions, Players, Roles, Teams, and Characters.

This league of legends database will be used to show and highlight the relationship among the entities, Region, Teams, Players, roles, and characters. This will allow us to update, and edit as we see the players progress through the world champion series 2019. The website will access the database and showcase through a search and filter system the relationships among the entities.

The database along with the relationships described further below.

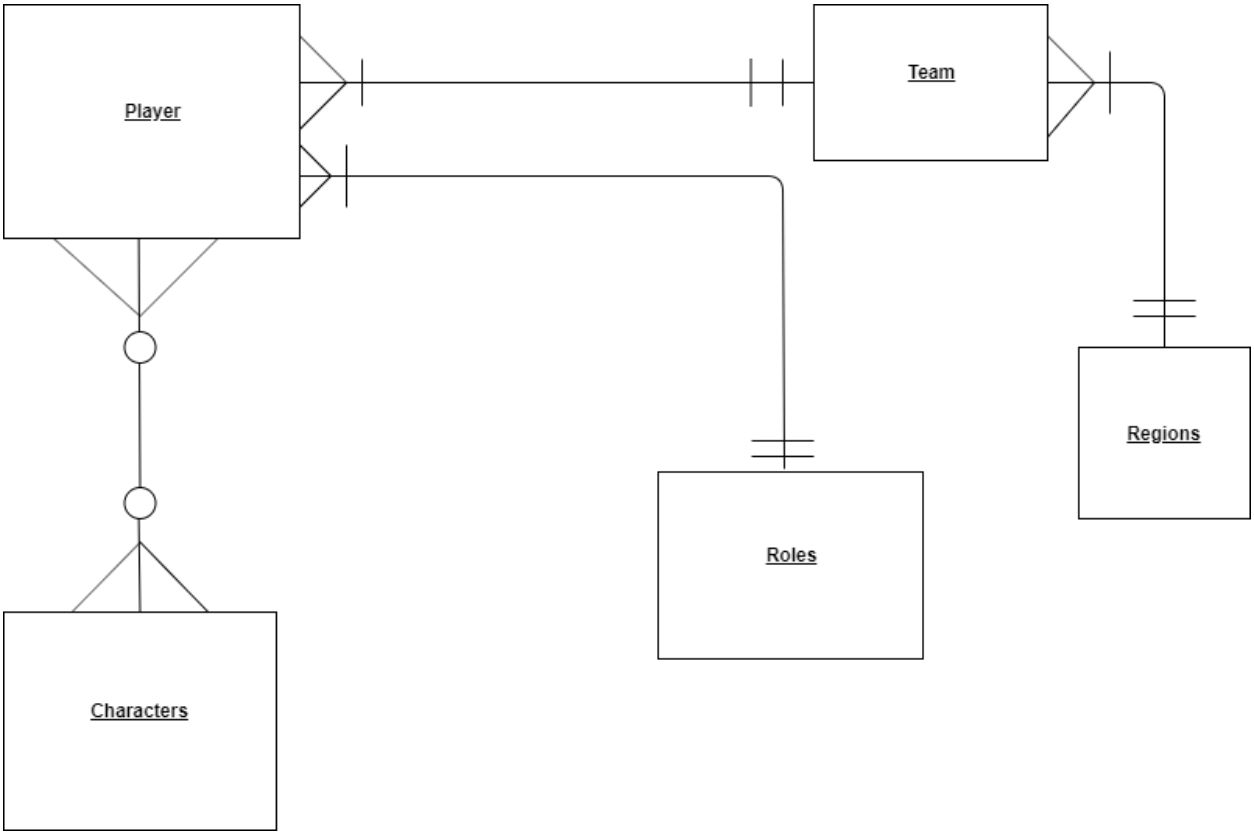
Entity	Attribute	Data Type	Miscellaneous/Constrains or relationships
Region	Region ID	Primary Key INT Type	
	Region Name	VarChar(128)	
	Team_ID(s)	INT TYPE	Foreign KEY
Player	Player ID	Primary key Int type	
	inGame_Name	VarChar(128)	
	roles_id	INT Type	Player can have one role ; FK
	team_id	INT Type	Player can only have one team; FK

Roles	Role ID	Primary Key Int type	
	role_name	VarChar(128)	Top, mid, jungler, ADC, Support
Team	Team ID	Primary Key INT type	
	team_name	VarChar(128)	
	player_id(s)	INT Type	Player ID; FK
Characters	Character ID	Primary Key INT Type	
	character_names	VarChar(128)	
player_Character	player_id	Int type	Foreign Key
	character_id	Int type	Foreign Key
player_roles	player_id	Int type	Foreign Key
	roles_id	Int type	Foreign Key

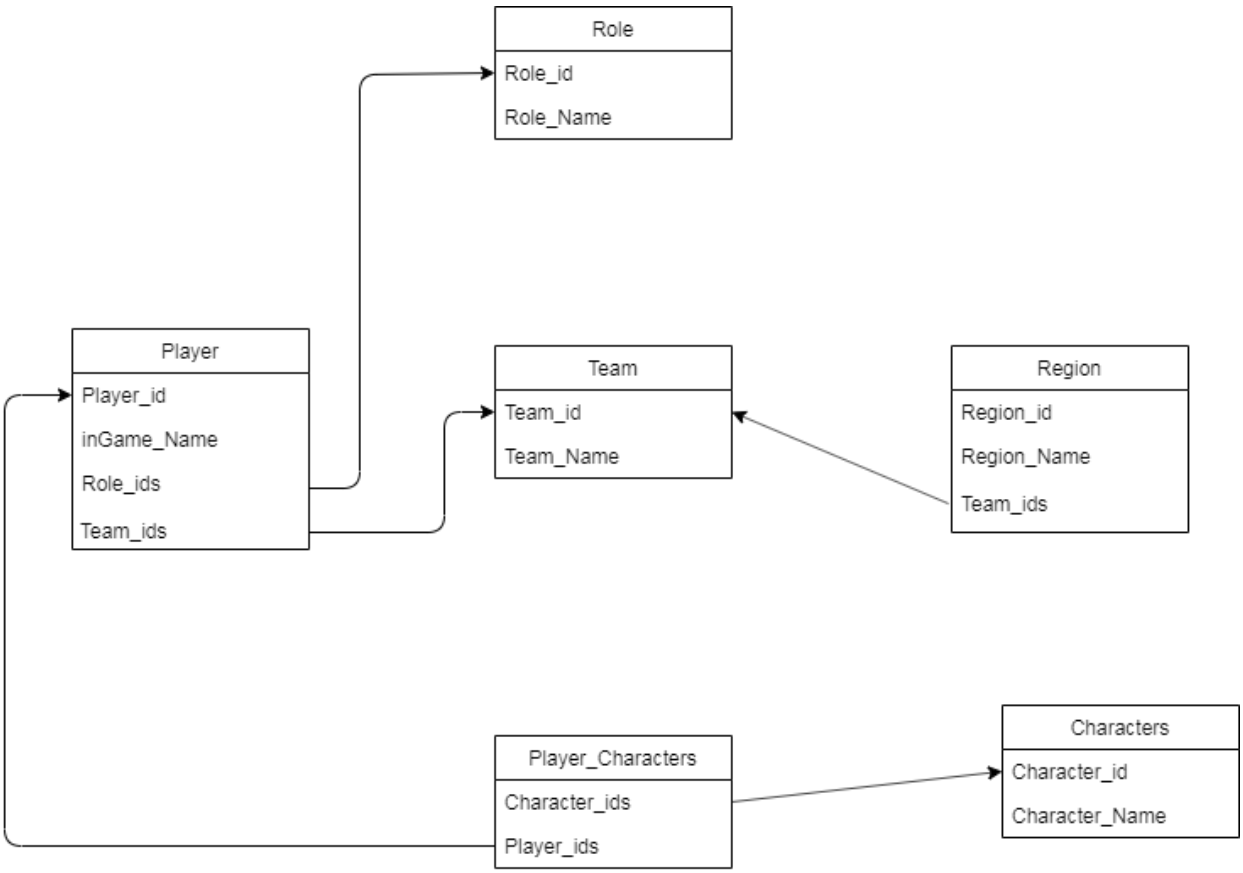
Relationships:

Relationship type:	Entities relationships
One to one relationships: ● One player can only have one role ● One player can only have one team ● One team can only have one region One to many relationships: ● One role can have many players ● One team can have many characters ● One region can have many teams Many to many relationships: ● Players can have many characters ● Characters can have many players	Players: ● Players can have one role ● Players can have one team ● Players can have many characters Team ● Teams can have many players ● Teams can have one region Roles ● Roles can have many players Characters ● Characters can have many players Region ● Regions can have many teams

ER Diagram-



Schema-


Role
Role_id
Role_Name
Player
Player_id
inGame_Name
Role_ids
Team_ids
Team
Team_id
Team_Name
Region
Region_id
Region_Name
Team_ids
Player_Characters
Character_ids
Player_ids
Characters
Character_id
Character_Name