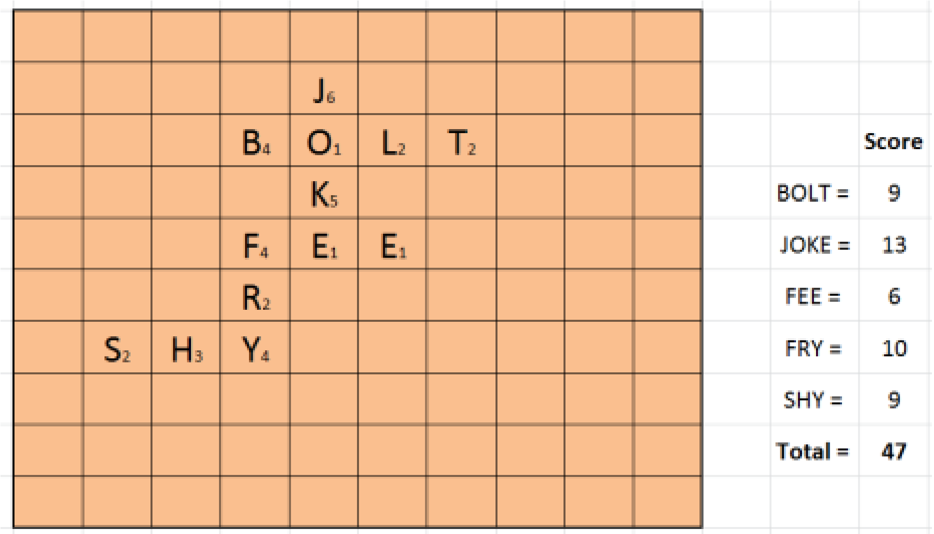
Senior Capstone Design in Computer Science
Project Description

Project Title: WordScuffle Game WebApp WordSCUFFLE	
Sponsor Information:	**Barbara Jenkins,** **NAU Budget Office, Business Analyst** Barbara.Jenkins@nau.edu

Project Overview:
With over 13.8 million people projected to be diagnosed with Alzheimer's disease by 2050, more and more people are looking to gaming apps to help improve their memory and maintain brain health. For example, Luminosity, a popular braining game is used by over 60 million people worldwide. Word games in particular help to improve vocabulary, word recall, and concentration…in addition to being fun. One of the most important motivating features of successful games in this arena is their social features, i.e., allowing multiple players to participate in a shared gaming experience online.

The goal of this project is to create a word game web application for a new game called "WordScuffle" that is fast, fun, engaging and mentally stimulating. Unlike some similar offerings currently available, Wordscuffle is based on a concept that allows more fair head-to-head competition, as all players in a round are given the set of building blocks to work with, and must each work to produce the best possible outcome from this common starting point. WordScuffle has some overall similarity to Scrabble, in that it is based on players putting together words from a set of given letters and calculating scores based on the words produced. The similarities end there, however, as Wordscuffle has been carefully designed to create a controlled competitive environment that uses specially designed letter sets to promote maximum word construction and mental exercise. The basic outlines of the WordScuffle concept are as follows:

- The game can be played individually or in a group of players (which might include a computer game agent).
- A specially designed game generator is used to provide all players in the game with 13 letters. All players receive the same 13 letters; they are the basis of that game round. The game generator uses a proprietary technique developed by the client that ensures that any 13 letter set of letters produced by the game generator is capable of producing a maximum number of assembled words. This keeps frustration low and increases user satisfaction.
- Like Scrabble, all letters have point values associated with them. Each player starts with a blank 10x10 grid and can drag and drop letters to form words, seeking to maximize the number of points. Note that each player has their **own** 10x10 grid; the grids of other players are not visible to a player…until the end, when scores and results are compared.



				J_6					
			B_4	O_1	L_2	T_2			
				K_5					
			F_4	E_1	E_1				
			R_2						
	S_2	H_3	Y_4						

	Score
BOLT =	9
JOKE =	13
FEE =	6
FRY =	10
SHY =	9
Total =	**47**

- During game play, the application manages all of the action, validating words as they are produced by each player on their board, and showing running scores totals for all players.
- Various game variations are possible, e.g., establishing bonuses for number of word intersections, word length, etc. The client will provide several model, and the team is encouraged to suggest other interesting options.
- At game termination, scores are compared, and players in a game may view the boards of other players to see how they managed to produce their scores.
- In the default game, players have four minutes to complete their grid, but other variations could be supported as well, e.g., a “race” where play continues until some player reaches some agreed on score threshold, or some other time limit is agreed and set by participating players.

The web application must allow users to register secure user accounts on the web portal, and should provide them with tools to track their play history and scoring performance over time. It should also allow various forms of team formation and team play, possibly including tournaments and other featured events. The design team is encouraged to work with the client in exercising their creativity to suggest further features and capabilities.

The web application must also be designed to accommodate future implementation of a monetization scheme. In general, the idea is that unlimited solo play should always be free, but that participation in the online community of players could require payment. The simplest example would be a paid registration for group play; a more sophisticated approach might involve having a running balance of “tokens”. Participating in group games might then cost a certain number of tokens…with some gaming groups (e.g. elite-rated player groups) costing more tokens than others. Tokens could be purchased…or won by scoring high in a play round. Again, the team is encouraged to contribute their creativity in helping design an attractive model here.

Key features of the product are therefore as follows:

Phase 0: Minimum viable product:

- A modern, secure Web2.0 application, deployed on a platform to be determined as part of design (deployment on a virtual server, e.g., AWS, preferred).

- Administration of site must be supported by a well-designed GUI suitable for a non-technical site administrator.
- Fully functional game play, with nice drag-n-drop board interface and functioning word validation and scoring.
- Supports both individual and group play in some way.
- Provides integrated instructions on game play.

Phase 1: a complete and useful application

- Ability to easily modify the proprietary game generator, e.g., to add new elements to it as they are developed over time by the client.
- Some mechanism (e.g. the token model) for future monetization of game participation.
- Ability to set the language for gaming groups or for individual games, i.e., switch dictionaries driving the game.
- More sophisticated mechanisms for group play, e.g., "gaming tables" which are just groups of players that are competing against each other. Might be restricted by invitation, or by your evolving skill rating.

Phase 2: Truly fabulous

- Ability to set up and deploy gaming tournaments. Players join a tournament, a play bracket is organized, and the tournament is run. Must accommodate differing time schedules of players around the globe.
- Provides a computer-based game agent. The idea is that the computer could show you other (presumably better-scoring) solutions to help you learn.

The team will further flesh out requirements in collaboration with the sponsor, and will explore technological options during the fall requirements and design phase.

Knowledge, skills, and expertise required for this project:

- Experience and interest in programming languages and frameworks used to create game web apps in a modern Web2.0 platform.
- Knowledge of mobile devices/tablets, as well as capability to create an appealing display interface on one.

Equipment Requirements:

No special equipment is required beyond a standard development environment.

Software and other Deliverables:

- A fully functioning WebApp that implements the complete functionalities outlined above. Installed and tested on a platform to be determined.
- User Manual for site operation and maintenance, designed for non-technical client.
- A strong as-built report detailing the design and implementation of the product in a complete, clear and professional manner. This document should provide a strong basis for future development of the product.
- Complete professionally-documented codebase, delivered both as a repository in GitHub, BitBucket, or some other version control repository; and as a physical archive on a USB drive.