

COuntdown

Applied project and minor dissertation for Bsc (Hons) of Science in Computing in Software Development

April 2019

Authors:

Thomas Duffy

G00339581@gmit.ie

Marian Ziacik

G00340481@gmit.ie

Supervisor:

Martin Kenirons

Table of Contents

[*Acknowledgements* 2](#_Toc534969412)

[Introduction 3](#_Toc534969413)

[Links 3](#_Toc534969414)

[Running the application 4](#_Toc534969415)

[Methodology 5](#_Toc534969416)

[Technology Review 6](#_Toc534969417)

[*Conclusion* 7](#_Toc534969418)

[*Learning outcomes* 8](#_Toc534969419)

[*Future Investigation* 9](#_Toc534969420)

[*References* 10](#_Toc534969421)

*Acknowledgements*

Introduction

The main aim of this project was to showcase skills we have learned over our 4-year course in GMIT.

The goal was to create a three-tier application which includes a front-end, back-end and database. The project uses JSP and Java Servlets to display the web page to a user, the user interacts with this webpage and the servlet uses Java to run the logic of the game and then return data back to the servlet which sends that data onto a JSP page to be displayed to the user. We use tomcat server to serve up the web pages locally, we also have deployed the app to Amazon server where it can be available to access from anywhere.

When the user enters a word, we use a python script to make an API call to the Oxford English dictionary to check the word is a valid English word before allowing the player to score points.

If the python script finds the word in the dictionary, then Java continues to score the letters used in the word.

At the end of the overall score of the player is sent to a database so we can keep a record of who has done the best. There is a page in the app which will display the top 10 players who scored the highest.

We will cover this in more depth in the technology section.

Notes: Amazon Idea:

In order to deploy the app to amazon servers we could install it the same way as we use it locally i.e install eclipse, install mongoDb, install python, install java.

Or we could generate war file and execute it in command prompt on amazon server. The war file contains the web application that can be deployed on any servlet/jsp container. The .war file contains jsp, html, javascript and other files necessary for the development of web applications.

Limitations

Known bugs

Technologies used:

Spring boot

Java 8

Junit4

ReactJS

WebPack

Bootstrap

Yarn

CSS

Html5

JavaScript

MongoDB

Maven

Git/GitHub

Docker

Research:

The original idea was to use spring boot java server that would connect to mongo DB, which in turn would talk to a React JS app as the root resource that connects to the HTTP API as a user. This idea was abandoned because we had our group reduced from 3 members to 2 members, and when we started coding the project we realised that we that we needed to spend a lot of time getting used to these technologies and without the third member that was proving very difficult. We have a branch in our GitHub which has a working basic spring boot and react app. Our problems were mainly getting these technologies working with our main game which was coded in Java.

So we moved onto using JSP and Java servlets to display web pages and a Java backend that does the logic for the game. Then we included a Python script that makes a call to the

We then use

Spring boot java server connected to mongo DB which talks to a React JS app as the root resource that connects to the HTTP API as a user.

Scrabble online is a HTTP web API that allows its users to create and play games of scrabble via HTTP requests.

There is a react JS front end served as the root resource that allows you to play and start new games while also keeping track of all in-game progress.

Java Server

Provides various HTTP endpoints which facilitate managing scrabble games programmatically via HTTP

React front end

React JS front end allows users to play and manage their ongoing scrabble game via a convenient web application.

System requirements:

Standard Modern web browser such as Chrome, Firefox, Safari and Edge etc.

Links

GitHub URL : <https://github.com/DuffyTJ89/CountdownProject>

<https://www.tutorialspoint.com/jsp/jsp_environment_setup.htm>

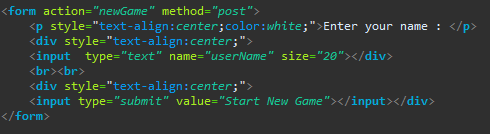
How the application works

Countdown is a web app which uses JSP and Java Servlets to talk with a Java backend which does all the logic of the game. The servlets are used to call methods from the classes in the Java backend and use the results to pass data onto a JSP page to be displayed to the user. This data which is displayed to the user will either be used to progress them onto the next step of the game or display the end results of their game to them.

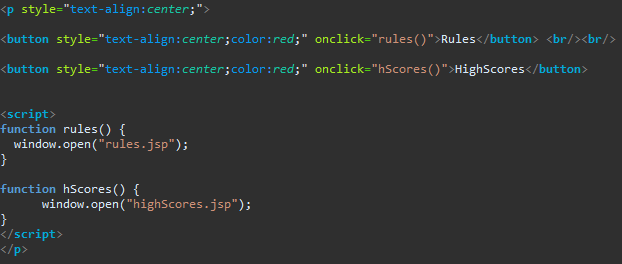
Using Tomcat to server the JSP pages, tomcat uses a default setting where it looks for an index file to serve after staring up. We use this index file as the first page of our web app.

The Index page displays to the user a textbox in which they are prompted to enter their name. This name will be passed to the Java servlet and the set as a session variable to be used throughout the application. We will pass this name along with the score the user gets at the end of the game to out database so we can keep track of all the users who have scored the best.

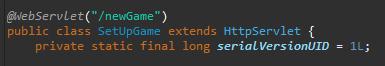
When the user clicks submit on their name the first Java servlet (called SetUpGame.Java) will be called by an action on the button seen in code below.



There are also links to a JSP page which displays the rules of the game and a JSP page which shows the high scores currently stored in our database



SetUpGame.Java is our first servlet. A servlet is a class which responds to a particular type of network request - most commonly an HTTP request. action=*"newGame"* method=*"post"* looks for a servlet to match the newGame action.



In this servlet we make the first of our calls to the Java backend.

Running the application

Methodology

Technology Review

Talk about spring boot and how it was set up

Maven and all the dependencies

Yarn, we used but removed it, why.

NPM and node.js, why we use. JavaScript runtime built on chrome’s V8 javascript engine.

Talk about babel and webpack and webpack cli, bundle.min.js.

HTML, CSS we created a JS folder separately.

JSP

JavaServer Pages (JSP) is a technology that we decided to go with in the end. We based our decision on the following reasons

* Knowledge and experience with HTML
* Knowledge and experience with Java
* Documentation was clear to understand
* The game was programmed in Java when we were testing it out so working with JSP would be reasonably straight forward
* We had experience using Tomcat to serve up pages before thanks to a pervious module in this course

JSP is similar to PHP and ASP, but it uses the Java programming language. To deploy and run JavaServer Pages, a compatible web server with a servlet container, such as Apache Tomcat or Jetty, is required.

*Conclusion*

We selected scrabble as we wanted ourselves and make use of all the skills we have learned during our time in GMIT. As the project progressed, we realised we needed to experiment with technologies and research them to figure the best way of implementing our idea.

The technologies we settled on where the ones we found were best suited for our projects. React and Node.js to develop the backend instead of Spring boot. The reason for this was needing to restart the server any time a change was made to the UI slowed us down greatly.

This is a project where it is hard to know where to finish as you can almost always improve it in some way. In the end we decided to stop at

*Learning outcomes*

*Future Investigation*

*References*