LoLArtemis.GG
Project Proposal

Abhiteja Joginipally
Aditya Vaderiyattil
Dhonovan Hauserman
Andrew Chang
Callum Moore
Jimmy Ly

ALL CHANGES MADE TO THIS PROPOSAL FOR FINAL SUBMISSION IS IN RED

GitHub Repo: https://github.com/ajoginipally/artemisgg

Overview

Callum Moore 12-10-15

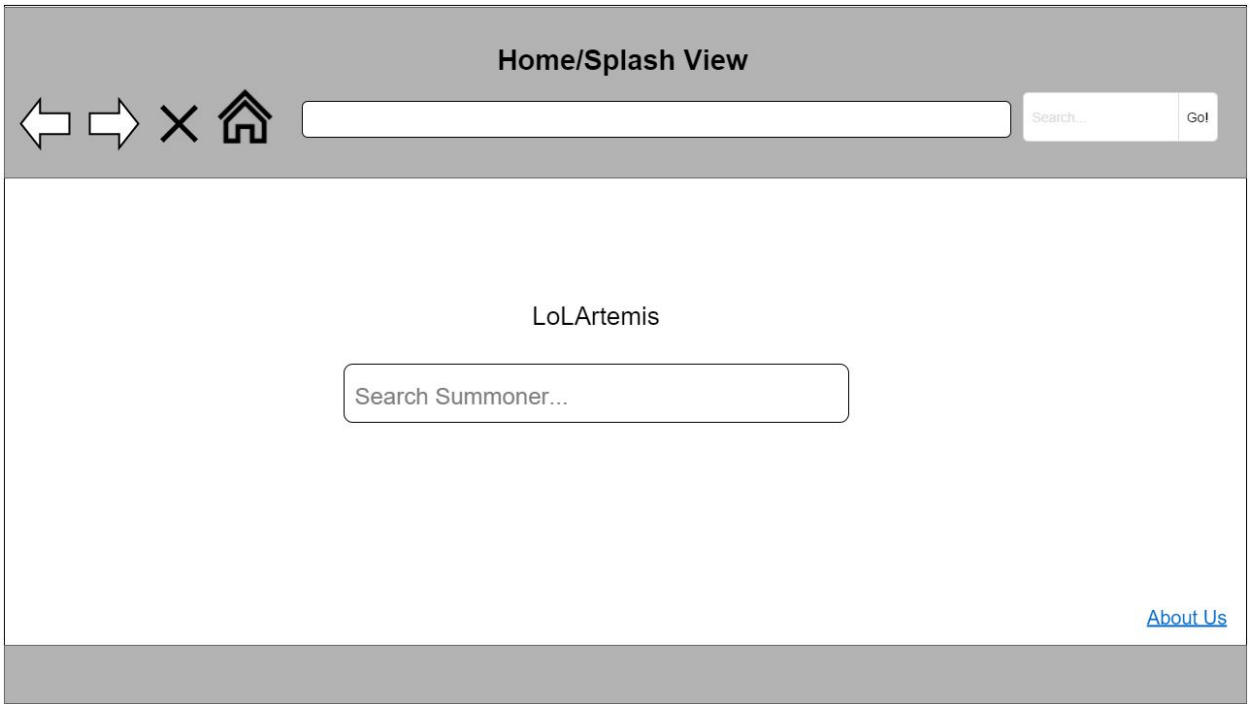
Artemis is a web based application aiding players in the online game, League of Legends. The application extracts data from previous matches of the enemy jungler to find relevant information on champion preferences and gank selection. These calculations are done with the League of Legends API. Consuming the data on the Artemis homepage, our user can adjust their play accordingly. This is the fundamental goal of Artemis: to give our users an advantage over the enemy jungler. The aim of Artemis deviates it from other supplemental League sites. Too often developers over engineer products. League sites often have superfluous data not useful to the user. For that reason we focused our product on pertinent data about the enemy jungler. The benefit of an application based on League of Legends is we only need to advertise to League of Legends users because there are millions of them. For this reason unlike other products we target

users who will likely be interested in Artemis. We do not need to get users interested in League of Legends; we need to get League users interested in Artemis.

Front-End Design

Jimmy Ly 12-11-15

We utilized mostly AngularJS and some sass for the front-end. We have these views: home, info, login, register, and an about page which is an exclusive admin page.



Welcome to ArtemisGG

Enter summoner name

Our home page represents the very essence of our web application allowing the user to search for a league account and their respective champion usage. There is a search bar that dynamically searches for the user's champions he used the past several games. The search button is responsible for signaling the backend to calling the league api.

Info View

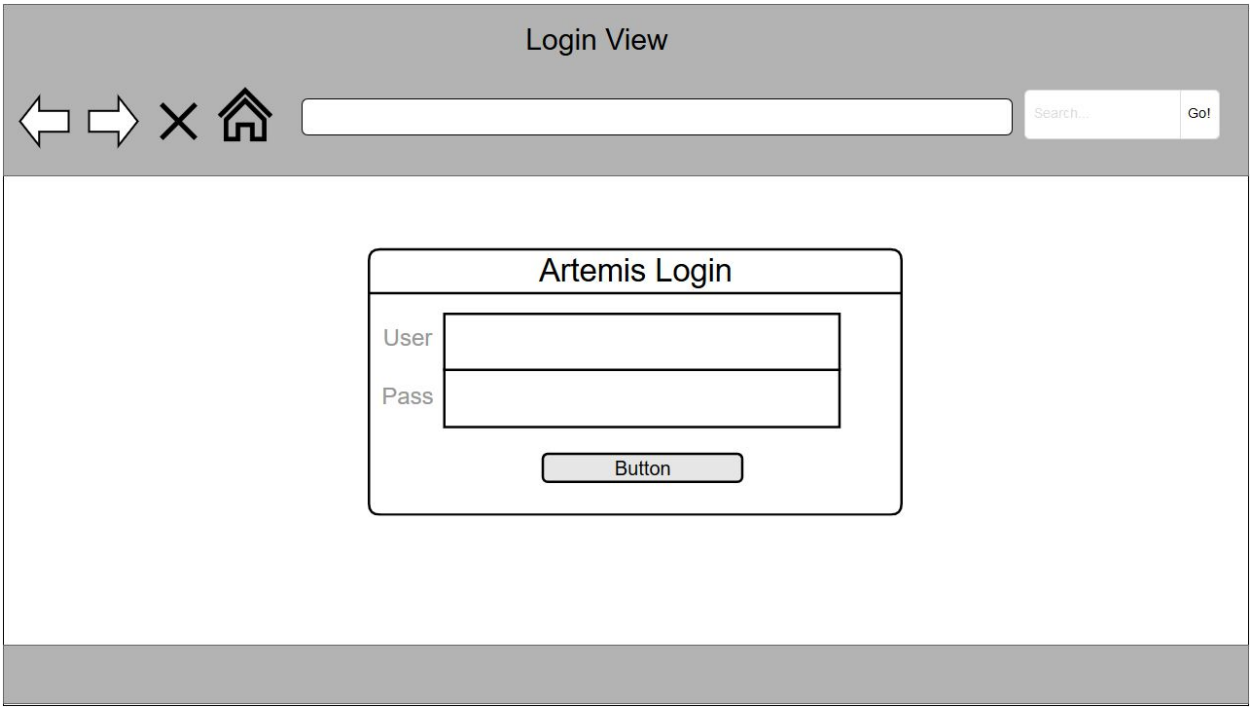
Search... Go!

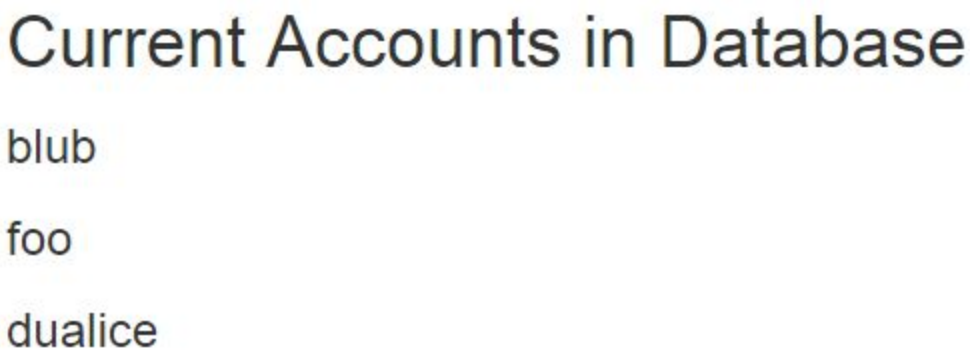
<Summoner> Jungle Information:

Chance of Ganking till 5:00		Chance of Ganking from 5:00 to 10:00		Chance of Ganking from 10:00 to 20:00	
Top:	75%	Top:	40%	Top:	10%
Mid:	20%	Mid:	15%	Mid:	60%
Bot:	5%	Bot	45%	Bot	20%

About Us

The info page tells whether the likelihood of someone is jungling based on their role on several past games. We also display pictures for the champions (there are some issues with some of the pictures due to naming conventions of the library we used compared to actual). Users don't need to register to utilize our application but they certainly can become a member if they feel the need.





This gives them access to some extra functionality that we were going to implement but we did not get to. The login is primarily for admins be able to access the admin page called user, to be able to perform basic CRUD operations on the website. There is a login and logout button which is used to maintain sessions.Admittedly, we only got it to do read operations from database at the time of this proposal. Furthermore, anyone who has registered can login to show their appreciation by using the app. The login button asks the database whether if the user exists and

if they do they are logged in. We intended to record and display their activity, but we were not able to add the extensive functionality in time.

Back-End Design

Jimmy Ly 12-11-15

The back end design consists of several parts that all work together to dynamically generate content for the front end. Passport handles most of the user authentication, specifically passport-local and passport-local-mongoose. Passport-local-mongoose enables us to connect to our mongodb database and use mongoose schemas for the purpose of authentication. Passport reads through the username and password field in our user objects and authenticates them in registration and login forms. The register and login routes are defined in our api. We also have a get request that pulls the list of users from the database to update our user list page. In order to maintain sessions and cookies we used passports built in middleware that piggybacks off on express session and cookies. The index.ejs generates the client side states in it's body. Finally our server.js file requires all these other files and creates the server.

What is League of Legends?

Dhonovan Hauserman 10/04/15

League of Legends is a fast-paced competitive online game played standardly with 5 players on each team. The game involves a huge battle-arena with each team's base in opposing corners. The objective for each team is to destroy the enemy base and protect their own. There are other modes that only have 3 players on each team and modes that are more like mini-games not part of the core game. The application focuses on the standard 5 player mode and none of the other modes as it has the largest player-base by far. In this mode one person on each team goes to the top side of the map, one person goes to the middle of the map, two people go to the bottom of the map, and one player roams around the map the entire game. This fifth person is known as the jungler who helps his team against the enemy in the same area of the map. A surprise attack by the jungler often leaves the enemy dead and forcing them to respawn in base and begin the long journey back to their area of the map and losing lots of time.

What does the app do?

Dhonovan Hauserman 10/04/15

We are creating a League of Legends application called Artemis, which enables one to extract data from previous matches of the enemy jungler to find their most frequent jungle pathing and ganking routes. This will allow the user to watch out at specific times and prepare for high-danger time frames, avoiding a likely death. This application can also be used to look at professional junglers to see where and when they usually gank to help the new junglers if they are not sure where and when to gank.

Team Organization

Abhiteja Joginipally (github: ajoginipally)

Client Side Developer UX and UI Designer

Abhiteja will work on developing the client side interface. In addition, he will create a rich UX environment and a UI enabling visitors to enjoy their visit to the website. Will implement the AngularJS framework, using Bootstrap to scaffold the application for mobile use. Abhiteja will write unit tests in Karma for the application.

What I ended up doing:

I worked on a full stack development and helped lead the team. I enabled a better understanding of Angular and the overall connection between the separate parts of the application. My biggest struggle was the client side interactions of http statuses and cookies. I primarily worked around the logic of a state based, single page application using ui-router.

Aditya Vaderiyattil (github: appu1232)

Full Stack Developer, but concentrating on the client side. Working in tandem with Abhi in developing client side interface and UI. Aditya will write unit tests. While Abhi deals with the UI and server access, Aditya will focus on the League API calls and accessing the data from the League servers, presenting it to the user.

What I ended up doing:

I ended up focusing much more on server side than client. I focused on League API as mentioned and I worked on figuring out the formula to determine where the enemy jungler would gank. It turns out, it is not easy to do frequent and successive calls to the API is not possible so I went

with a more conservative and admittedly less accurate approach. I also focused on documentation.

Andrew Chang (github: changity)

Database architect working on developing the database for the application. Incorporating the JSON files from the game data. MongoDB looks to be the ideal database software to use because of it's ability to directly insert data in JSON Format. Processing the League API calls and storing it inside the database and processing the data to be displayed based on the user.

What I ended up doing:

I ended up providing support to Abhi with Mongoose which allows us to use Mongo and Node together to establish our database easily and quickly. I filmed all of the footage for our commercial, coding, and website overview. I collected the audio for our clips, with help from Dhon. Editing together the audio and video files, I created the final videos.

Callum Moore (github: callummoore)

Project manager for Artemis. As a jack of all trades and master of none, Callum will help dissolve blockages on the sub-projects. In addition to his wealth of database knowledge and understanding of Javascript. After the initial database work is completed, he will likely move to client or server development; necessity depending. All the while keeping track of roadblocks, progress, and deadlines for the team.

What I ended up doing:

As the project manager for Artemis I focused a lot less on the technical side of things than I had hoped for. I was able to provide insight in some areas. Helping with and suggesting the implementation of passport.js for our user authentication. I had previous experience with passport from 320, software engineering. The project was done in angular. Angular was not something I was familiar with and as a result I had to leave most of the technical sections of the documentation for Abhi, Aditya, and Jimmy. Finishing what I could. Aside from editing and writing the documentation I looked into the league api for relevant queries. However, due to time constraints after a rocky project three, we refocused the app and data displayed on our homepage. In addition to the documentation, I wrote the script for our commercial and starred as the new user of Artemis.

Dhonovan Hauserman (github: donhouse)

Server Side Developer

Will write server code to listen to requests and fetch resources needed from our database and forward them to the client. This information is sent as JSON data and the communication done using http calls.

What I ended up doing:

I ended up focusing on where to obtain information from the League API to work with Aditya's new system for ArtemisGG since constant API calls is not allowed by Riot Games, and how to incorporate these database calls into our web application. Lastly I ended up working extensively on all of the videos, such as setting up the recording equipment and making sure each video was recorded correctly. This includes setting up the coding video, helping Callum with commercial script, and helping Andrew with the editing of the remaining videos.

Jimmy Ly (github: jimmyly)

Full Stack Developer but will primarily deal with server side.

Familiarize myself with the different libraries that deal with the server side. I will work with Node.js and Express(the middleware) and to work with the RESTful API. In the event the scope of the project makes the client-side harder than predicted and or if I have more time, I will assist Abhiteja on the client side interface. I will also work with Aditya since there is a lot of details we need to get through in the results of our preliminary research.

What I ended up doing:

Ended up helping more with the front-end than the back-end and pair coded with Aditya and Abhi. More specifically, I did some css and used AngularJS while trying to do some coding/debugging for Node.js or the server side. Making sure authentication worked properly took some time, but using the passport library helped a lot. I helped with some of the explanations for the documentation.

Disclaimer

Abhiteja Joginipally 10-14-15

This document is a preliminary sketch of the final product. Thus, this document should be viewed as such and should not be taken as final. Many edits will be made to this document over the course of development of the application. This enables a level of transparency between the team members and creates an environment that reflects the state of the application.

General Technical Specifications

Abhiteja Joginipally 10-4-15

The application must be built in a full stack manner with a server side, client side, and database. The server side will be written in Node.JS using Express as its framework. The client side will be written in Javascript using AngularJS as its framework. The client side will also have views and styles which will be written in HTML and CSS respectively. Twitter's bootstrap will be used to scaffold out the application and make it mobile ready. HTTP calls will be made to the League of Legends API to gather information about champions, games, etc. Most of the HTTP calls in the League of Legends API are GET requests to get necessary JSON game data. Data Dragon is another important tool to retrieve information about in game images and other static data. MongoDB is the database that will be used to store the data for the application. Gulp will be used to run tasks and automate the build to see how the application runs in a production like environment. Sass will extend the native CSS language and allow for more complexity in design. External modules for AngularJS will be used to for an easier and more efficient code base. Yeoman will be used to initially scaffold out the code base.

Dependencies, Libraries, and Languages

Abhiteja Joginipally 10-4-15

Official LoL API	Node.JS	Express	AngularJS
Javascript	CSS	HTML	Bootstrap
Data Dragon	Gulp	Sass	Angular Modules
MongoDB	Passport	ejs	cookie parser
morgan	body parser	connect flash	express-json
mongoose	serve-static	wiredep	jquery
restangular	angular-animate	angular-touch	angular-ui-router
bootstrap	jquery	angular cookies	angular filter
passport	passport-local	passport-local-mongoose	

Design Specifications

Abhiteja Joginipally 10-4-15

In order to create a pleasing UI and UX a grasp of the League of Legends aesthetic is highly necessary. To accomplish this, the team must play the game and keep an eye on how platform is set up. Placement of information and statistics on the game client as well as background art are crucial aspects that need to be researched. The team needs to also look at existing League of Legends web applications and collect tips for the most intuitive way to set up the application.

General Preliminary Research

Abhiteja Joginipally 10-4-15

In order to build the application there needs to be research done in a variety of areas. These areas of the project can be split into two categories. First, there needs to be an understanding of the technical specifications that the application entails. Second, research needs to be done on the League of Legends API and the game's mechanics.

Frameworks and Libraries Research

Jimmy Ly 10-4-15

Since Node.js and using Express is something we are still actively learning in class, Node.js and Express should take approximately about two weeks to properly make use of in the project. For AngularJS (those who are working or assisting with client side) will have somewhat of a learning curve for those who are not familiar with the framework. Since this topic does not seem to be covered in the syllabus, this could take about a few weeks to research to get familiar with the framework. A vital part in understanding AngularJS is a look into the MVC design pattern and how MVC frameworks are set up. In the case of REST API and JSON, this could take until the actual lecture to have a decent grasp on REST and JSON. Since we are still in the beginning stages, there is still much more research to be done on other frameworks to take advantage of, so timeframes are expected to change.

Specific Dependency Research

Node.js & Express

Jimmy Ly 10-4-2015

Node.js is fast in terms of deployment and implementation, the framework takes advantage of many libraries that allows for quicker development and a reasonable overhead. Node.js is lightweight and single-threaded resulting in a high degree of scalability. A feature of Node is it's non-blocking; instead of stalling for IO requests to be completed, it continuously runs IO requests one at a time making full use of a single core. However, it has to wait for the data sometime in the future. While IO intensive, CPU power is better utilized and thread is run in parallel due to the asynchronous nature of Node. Express is a minimalistic request handler allowing for the creation of hybrid web applications.

AngularJS & Javascript

Abhiteja Joginipally 10-4-2015

Using vanilla Javascript can only go so far. In order to easily create a rich dynamic application a framework should be added on top of Javascript. Google's AngularJS is a good choice to fulfill these needs. AngularJS allows for quick and expressive coding that can suit whatever needs are required by the client. Angular modules allows for extensibility and ease of use with other libraries.

MongoDB

Andrew Chang 10-4-15

In order to understand the data generated, The data will will need to be stored in the best way possible. Since the data is going to be received from the League of Legends API will be in JSON format, the MongoDB is a cross-platform document-oriented database that favors the use of JSON-like documents making the integration of data in certain types of applications easier and faster. Based on this, the MongoDB will be the database of choice for the application. The benefits of using MongoDB over other options is that it is schema less, the structure of a single object is clear, and the ability to index on any attribute.

Why use Artemis?

Abhiteja Joginipally 10-4-2015

League of Legends has a huge and highly competitive community eager to learn about game mechanics. The best way to learn and get better is through example; knowing how successful other players are with their characters and what they do to win their games. However, much of this information isn't readily available for gamers, who want an edge over their peers. The application provides players data about what pros are building and successful games that they've had with those builds.

Relation to Existing Apps

Aditya Vaderiyattil 10/4/2015

Riot Games, the developers of League of Legends encourage the community to create web applications for players to use and get any information that isn't readily accessible through the game itself (so long as it is not qualified as cheating). Riot's League of Legends API allows third-party developers to gain access to the League of Legends servers and data to display on their website. There are many well known websites that are actively used by the community. Here are examples of popular websites:

champion.gg	quickfind.kassad.in	probuilds.net
op.gg	lolking.net	lolskill.net

How Artemis Will Differ From the Existing Apps

Aditya Vaderiyattil 10/4/2015

All the existing web apps currently lets the user enter a Summoner Name (a League of Legends account) to get information on their past games. This helps someone who is loading into a game against an opponent and wants to know more about how good they are based on what characters they usually play (and if they are good on the character they are currently playing), what their ranking is (the game gives rankings to players based on their skill), and other information about their character. In our application we are striving to look at the person's play style rather than the character itself. Specifically we will focus on a certain role in League of Legends called, Jungle, who aims to roam the Summoner's rift, searching for allies to help and opponents to kill. Using the League of Legends API, we strive to get data on the Jungler that the user searches for and based on their previous games; determine when and where the Jungler will make their moves to give his team the advantage. A user can get access to this data and either be on alert or have

peace of mind when they are playing since they will have a good idea of where the enemy Jungler will be.

Conclusion

Callum Moore and Aditya Vaderiyattil 12-10-15

After programming and coding the functionality for Artemis; time would improve our final product. We are happy with the outcome of Artemis. It implements mongo, League of Legends API, CSS, html, passport.js, and Node. It gave us a chance to expand our skills from the class. However, with more time we could focus on our design, and experiment more with the league API. Although our ability to manipulate the League API is not essential to web development, more complex algorithms would set Artemis apart from competitors.

Our project was not without its successes. We were able to accurately query the riot League of Legends API parse and send that to our MongoDB database. In addition to fulfilling the functionality we were able to use angular. The implementation was more difficult using states, rather than views, but we felt it produced a seamless design. While our algorithm for gank likelihood was not as accurate as we hoped, it still is indicative of jungler tendencies. We were able to query for our algorithm and parse for the correct data. Artemis was a learning experience for our team, showing the problems encountered in the development process.

The League API restricts the number of calls and the frequence of the calls so our original idea of fetching past jungle ganks, kill, and deaths would not be possible. Instead, we came up with an alternative, less accurate way of finding the most likely to gank lane: We used the junglers last 15 games and looked at which lane they played in those 15 games. If the jungler played more top lane than any other lane, then the chance of ganking top lane would be the highest. In other words, the probability of ganking a lane is based on their experience playing the lane. This is not ideal, but it's easy to see how a jungler is more likely to gank a lane he plays more often.

As stated, the most difficult part of the project was determining how to use the League API to accurately fetch the data on the jungler and present information that could be relied on. In the end, we were not able to implement a champion filter or showcase popular junglers as the majority of our time was occupied by the problem of fetching the information and also because this would have lead to too many API calls. If we had more time, we would have liked to add these features to make the site feel like more than just a way to look up if you had a good chance of getting ganked or not. We wanted to be able to make it a site where someone could learn about ganking patterns based on pros or other summoners and could master a specific jungler by looking up common jungle paths for that jungler. However, we were able to implement the main feature that we advertised about the site and after testing it for our own summoner names and our friends', the algorithm does seem to be fairly accurate.