GitHub Username: manu120189

Social LoL

Description

Note: I have a friend who is enrolled in the Nanodegree IOS and for his last project he developed a fun app called "SociaLolApp", talking to him we think it would be fun I develop a version for the Android platform.

Problem:
Currently the only way to keep up with your friends performance in League of Legends is to log in to the game or 3rd party web pages.

Proposed Solution:
Design an app that let users search friends/players using the League of Legends API, Follow players all around the world checkin them to a favorite list, check the last matches played by your favorite players in the “History” view and check what rank are they, Know a little bit more about the game and his characters with the “Champions” view.

Intended User

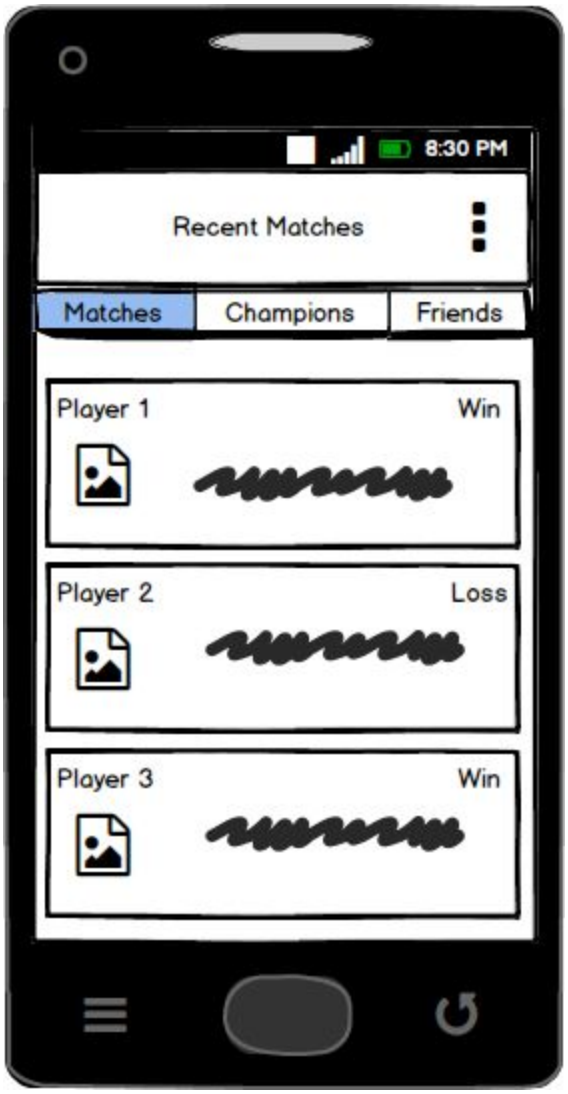
This app is mainly for League of Legends players.

Features

- The "History" screen where you can see the last matches played by your friends.
- The "Champions" screen where you can find all the available characters in the game to play with. You can select from the table the character you wish to know about an get some more detailed info such as abilities, history or tips.
- The "Friends" screen where you can find all of your added friends ranked by their game points and leagues.
- The "Search" screen where you can search both champions or friends and watch their profiles. Inside the friend profile you can add follow him as your friend in order to keep track of his last games.

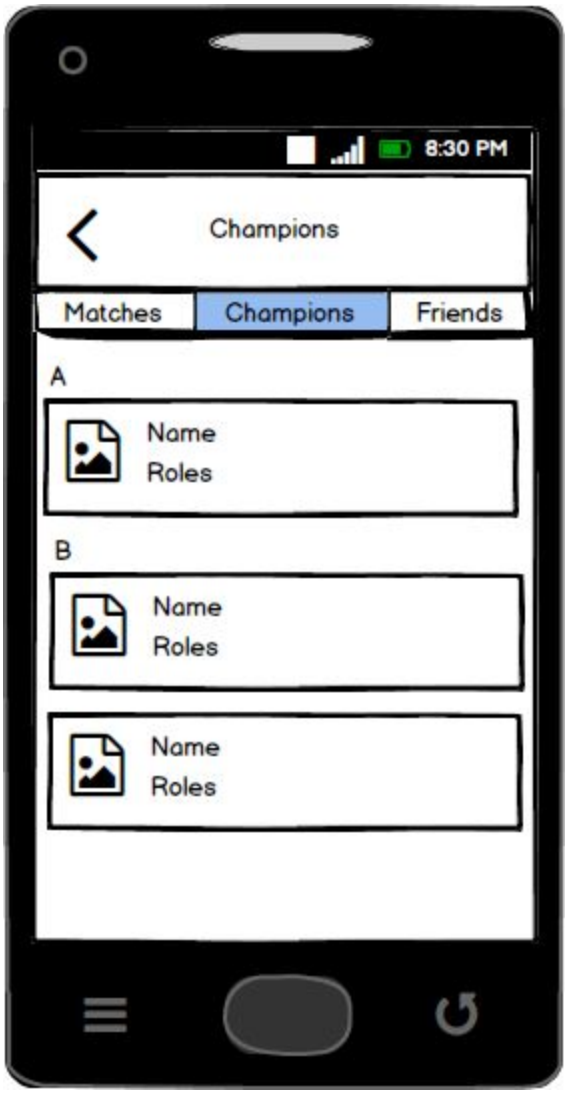
User Interface Mocks

Recent Matches Screen



Description:
This view is where the user can visualize the recent activity of his friends, after the splash view this would be the main view.

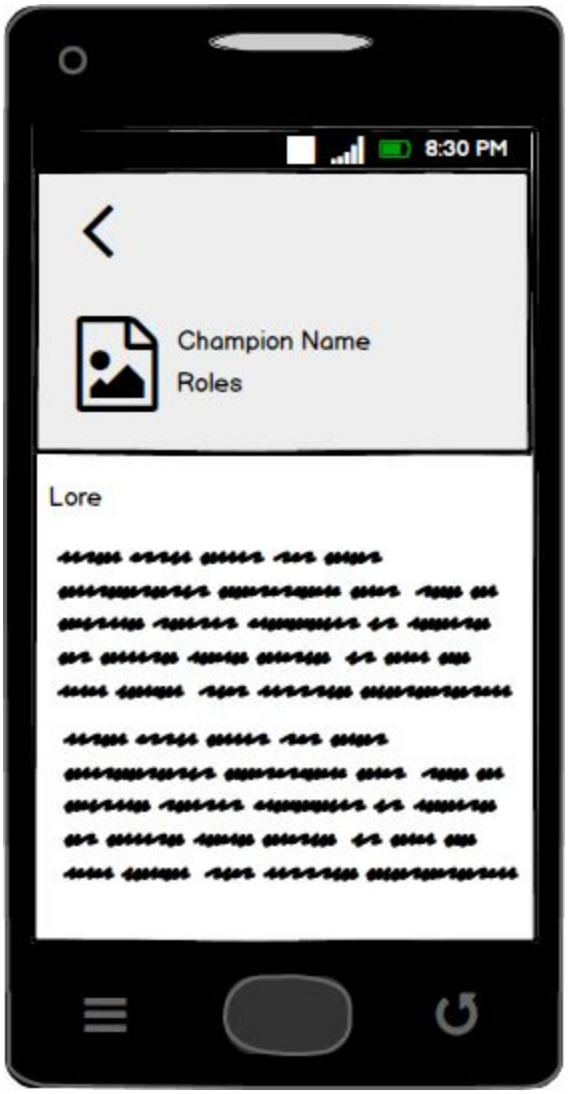
Champions Screen



Phone

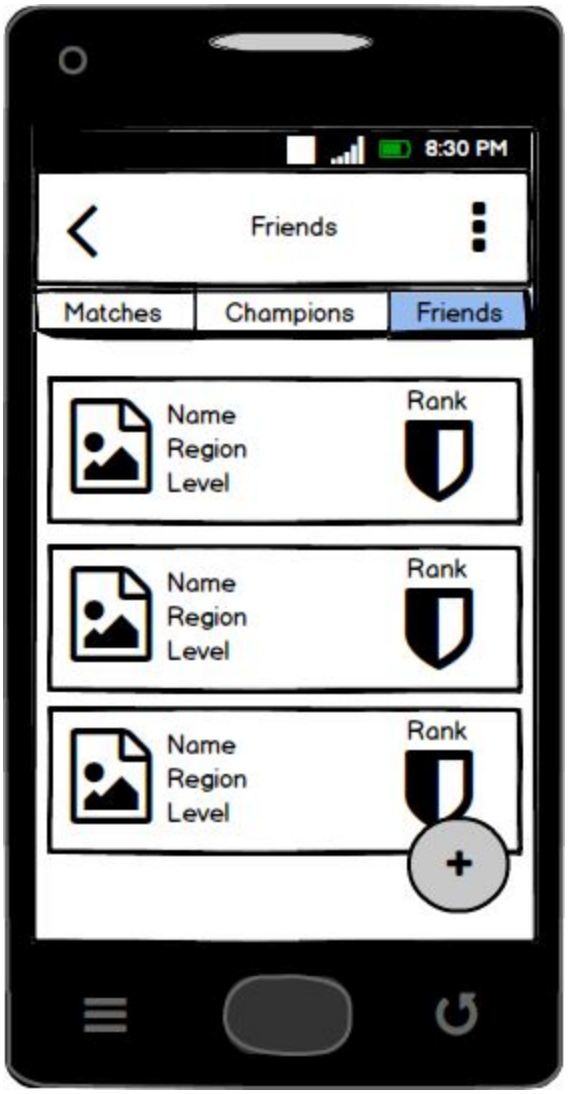
Description:
This view is where the user can visualize the different champions in the game, they can click in every champion and they would be redirected to a detail screen.

Champion Detail Screen



Description:
This view is where the user can visualize the Lore of the selected champion..

Friends Screen

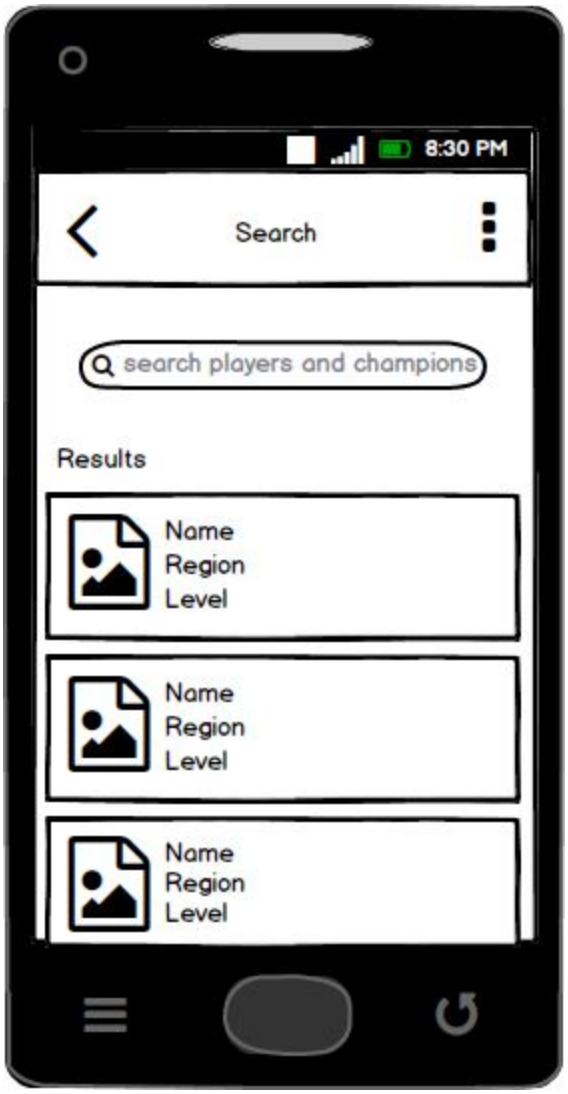


Description:
This view is where the user can visualize his friends list, here is like a favorite list of all the players the user follow, here we have the Fab button that redirects to a search view, if the user clicks in a friend would be redirected to a detail view.

Friend Detail Screen

Description:
This view is where the user can visualize the detail of a selected player in the Friends Screen/ Search Screen, if the user comes from the friends list the follow button would be hidden, in this screen we have statistics of the player, top champions he plays, and his current rank in the League of Legends system.

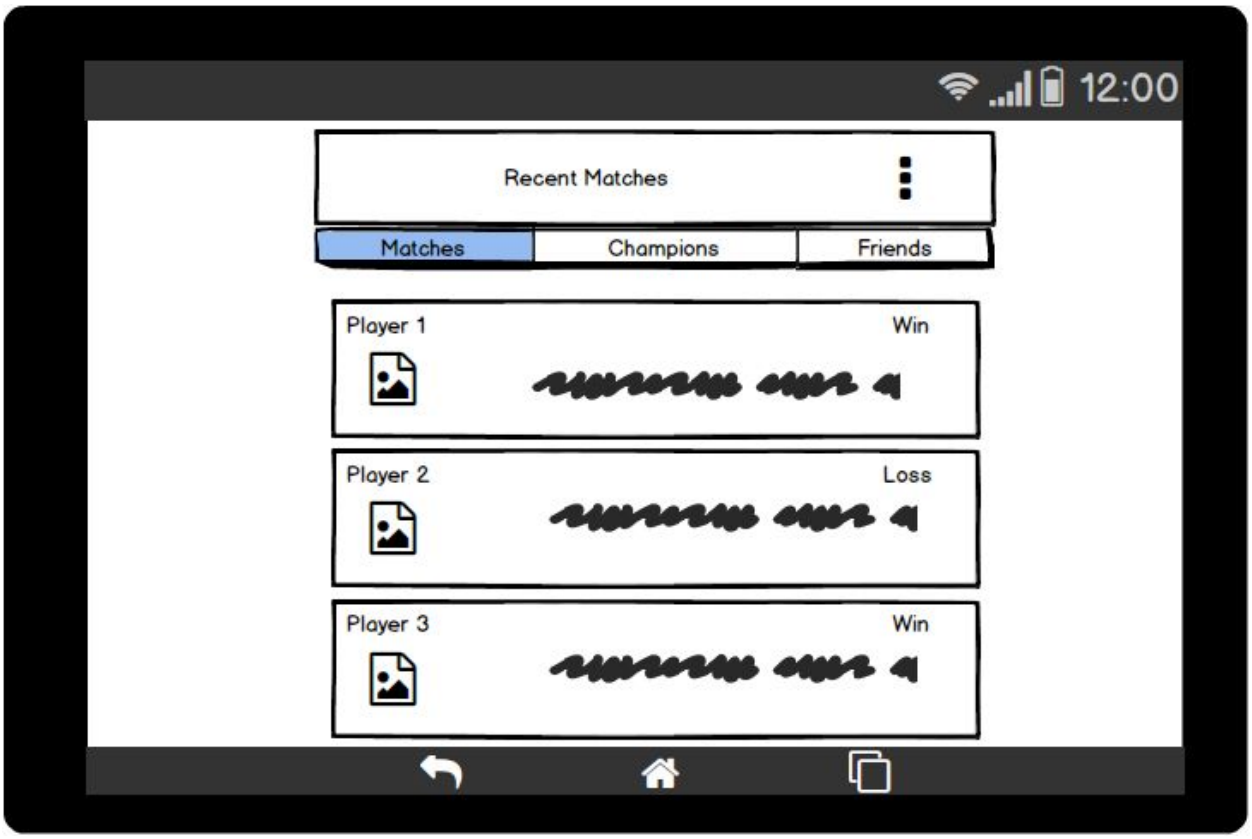
Search Screen



Description:
This view is where the user can visualize the the search section, here he can search a LoL player or a champion, if he clicks on any of those a detail view would be displayed,The details views displayed are the previously mentioned, **Friend Detail Screen or Champion Detail Screen.**

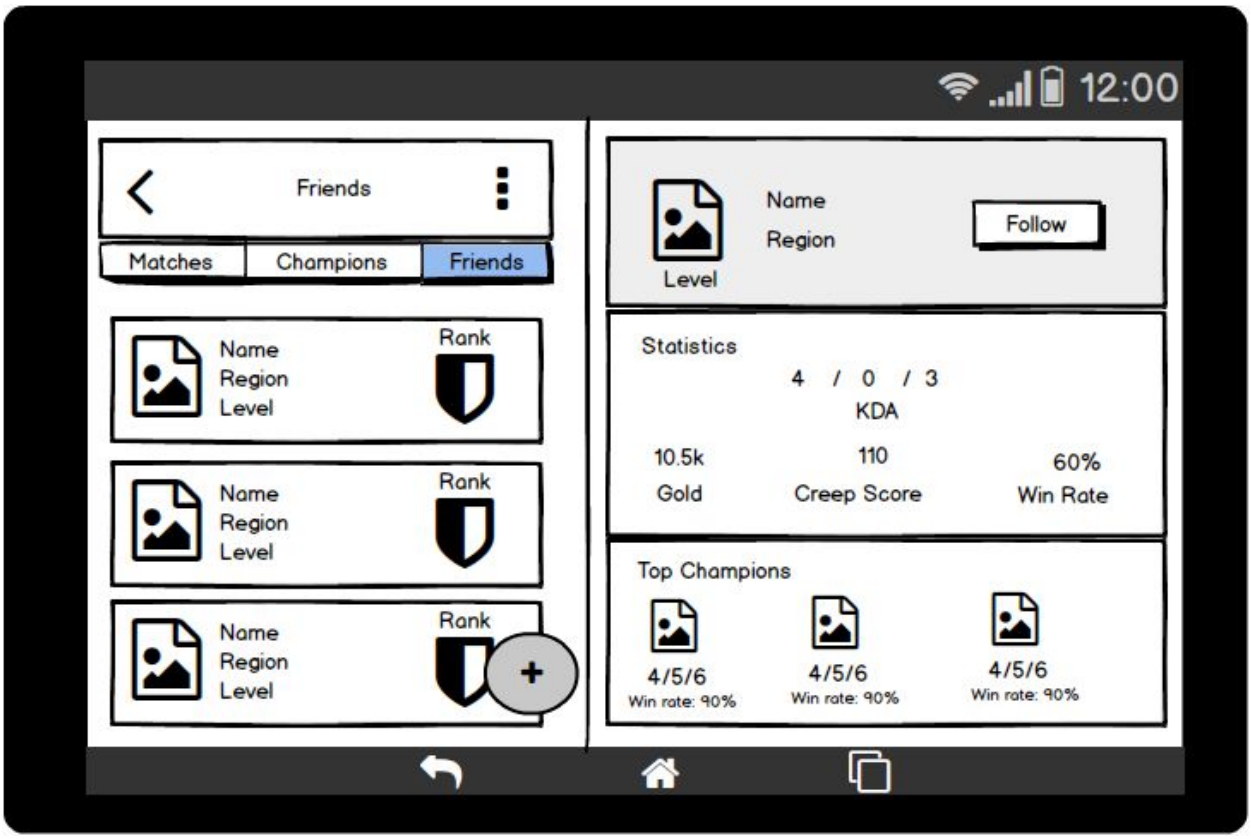
Tablet Mocks

Recent Matches Screen



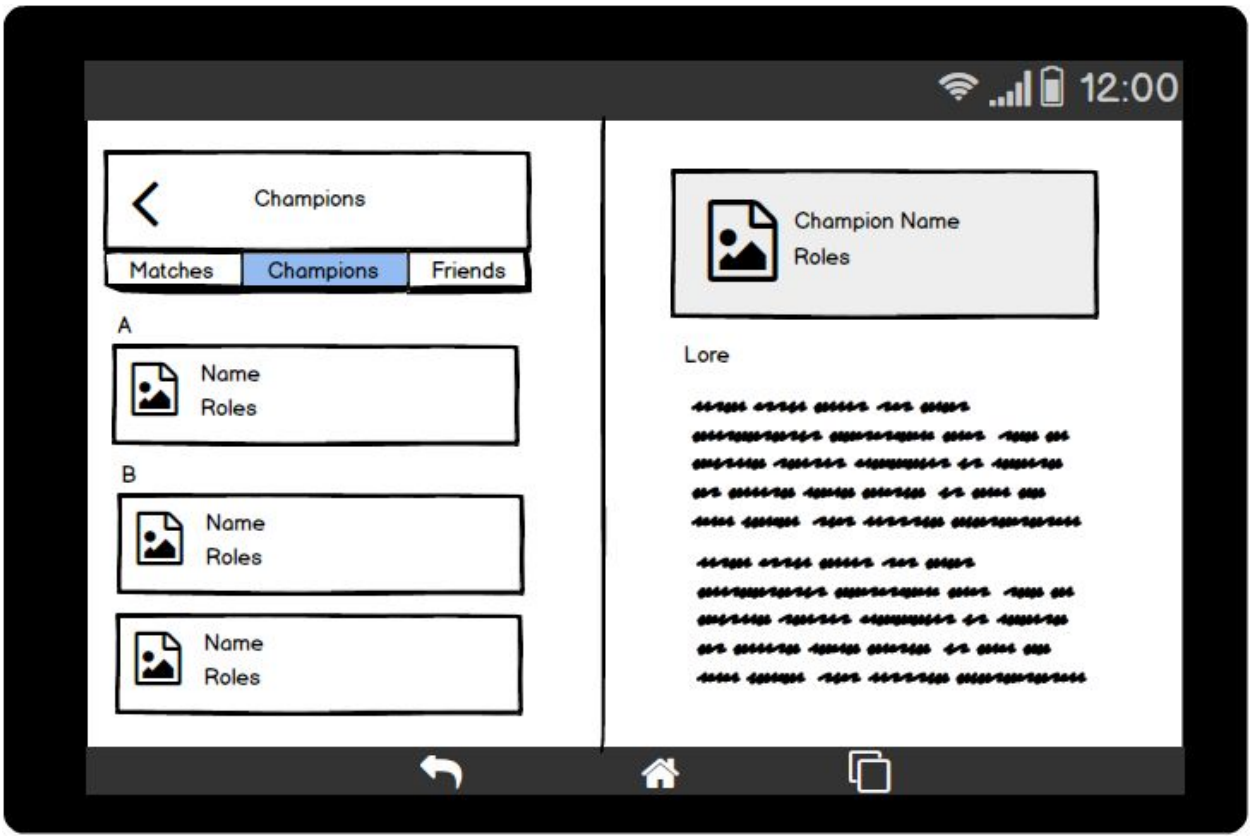
Description:
This view is where the user can visualize the recent matches of the user's Friends list.

Friends Screen



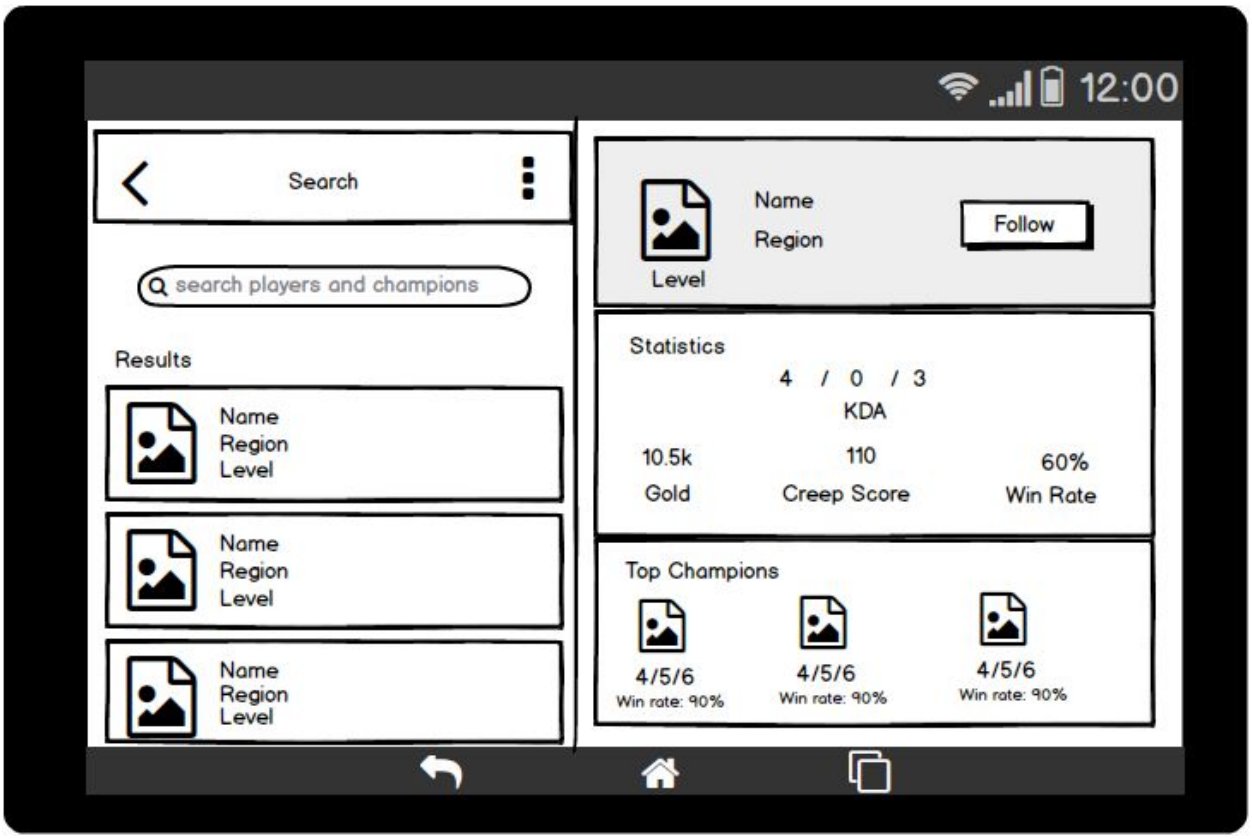
Description:
This view is where the user can visualize the Friends list, if a friend is clicked, the detail view would be updated/

Champions Screen



Description:
This view is where the user can visualize the Champion list, if a champion is clicked, the detail view would be updated.

Search Screen



Description:
This view is where the user can visualize the Search list, if a player/champion is clicked, the detail view would be updated according to the selection

Key Considerations

How will your app handle data persistence?

My data would be persisted in the phone local database using sqlite, and an ORM to handle the data like ORMLite.

Describe any corner cases in the UX.

For example, how does the user return to a Now Playing screen in a media player if they hit the back button?

Describe any libraries you'll be using and share your reasoning for including them.

1. Picasso, to handle images requested from the League of Legends API and load them to the UI.
2. Retrofit to handle with observables the calls to the LoL API
3. RXAndroid to implement async procedures in the app, also is compatible with Retrofit.

Describe how you will implement Google Play Services.

I think my app won't be using any of the GPS

Next Steps: Required Tasks

Task 1: Project Setup

- Create repository for the project.
- Create the Android project with target to the latest SDK
- Configure initial libraries, Picasso, Retrofit, RxAndroid
- Configure ORMLite to use in the project
- Check the League of Legends API and document for myself the Endpoint I'm going to use.
- Check a source of free images of League of Legends to use in my layouts.
- Configure access keys for the API in the gradle build file
- Configure flavors in the gradle builds file, Production and Debug

Task 2: Configure Access to API and DataBase

- Implement access to league of legends API using Retrofit.
    - Map all necessary objects from the json response of the API
    - Map all necessary endpoints and methods (GET,) that the app is going to use..

- Implement layer of DB access with ORMLite defining an interface of methods that the logic of the app will use.
- Implement an Picasso interface to handle Load of images in the app.

Task 2: Implement UI and Logic for Each Activity and Fragment

- Build UI for Splash Screen
- Builds menu of the app with tabs
- Build UI for MainActivity with the help of CoordinatorLayout and two pane view for tablets.
- Build UI for fragment RecentMatches
    - Create ListView for matches
    - Implement logic to retrieve the matches information from the API once every hour or if the user want to refresh it
    - Handle error in case of API is down or lost connection
- Build UI for fragment Champions
    - Create ListView for champions
    - Implement logic to retrieve the champions information from the API
    - Save the champion information to DataBase
    - Handle error in case of API is down or lost connection
- Build UI for fragment FriendsList
    - Create ListView for friends
    - Implement logic to retrieve the friends information from the API once every hour or if the user want to refresh it
    - Handle error in case of API is down or lost connection
- Build UI for fragment Search
    - Create ListView for search results
    - Implement logic to retrieve the search information from the API
    - Handle error in case of API is down or lost connection
- Build UI for fragment FriendDetail
    - Create view for the friend detail
    - Implement logic to retrieve the friend information from the API
    - Implement logic to “Follow/UnFollow” a player
    - Save information to database

    - Handle error in case of API is down or lost connection
- Build UI for fragment ChampionDetail
    - Create view for the champion detail
    - Get information from database, is not necessary to search this information every time in the API
- Implement logic for two pane view with the Fragments, FriendList/FriendDetail, Champions/ChampionDetail, Search/FriendDetail/ChampionDetail.