System-wide Requirements

Project

UARobotFight

Team

UARobotFight

Author(s)

name	id	email
Pedro Costa	49648	costa.p@ua.pt

Revision(s)

date	Who	What
March 5, 2013	Pedro Costa	All

Table of contents

1. Introduction

The main objective of this project is to deliver a game to be used in competition between UA students.

Since this solution consists exclusively on a NodeJs application and a PostgreSQL database, the system requirements are very basic.

2. System-Wide Functional Requirements

- Provide a mechanism to authenticate UA students (through the authentication service of UA (idp.ua.pt)).
- Let the user create a new robot, edit the code, using the game API, and try to fight against bots on editing page.
- Let the user fight other robots, random fight or selecting the opponent.
- Let the user other robot's code and use it to create a new one (fork).
- Provide information about a user profile, all robots their statistics (graphics charts).
- Provide historical results (fights).
- Show a fight replay.
- Provide information about the Top 10 Robots in the competition.
- Provide information about the user classification on the competition.
- Provide help information about the game API (Documentation).

3. System Qualities

3.1 Usability

The game is played on a web browser. So the user must be connected to the Internet and must be a UA student.

- Encourage the user to create a new robot and enter in the competition

3.2 Reliability

In very stressing situations, high affluence of users, attacks, etc., our site server is likely

to crash. Because is centralised on a unique machine and it depends on internal specifications of the technology used (NodeJS and PostgreSQL). But it is very easy to recover.

3.3 Performance

The whole system is very light, fast to respond to new requests. Is very quick and easy to startup or shutdown.

3.4 Supportability

To optimize the system, the Data Base could run in a separate machine. But even upgrading the hardware of the current machine could be enough.

The operating system is Linux.

To make the whole system work is only required to install NodeJS and PostgreSQL and run the make file. This means that, theoretically, could run in Windows.

The system has a log for support and maintenance.

4. System Interfaces

The system is a web site, which means, it's used in a browser. Possibly in a tablet browser also.

4.1 User Interfaces

The system user interface consists in several pages. Each with a specific role in the use case implementation and system functions.

4.1.1 Look & Feel

The system look and feel will be minimally consistent with UA. (green, using UA symbols, etc.).

4.1.2 Layout and Navigation Requirements

In the home page will be a informations about the number of users and robots and the last fights, to let the user replay them.

4.1.3 Consistency

The site have same static areas through all pages. For example: header with a menu; footer with a simple look and information about the developers.

4.1.4 User Personalization & Customization Requirements

The system look will be the same to all user and it will be not possible to customize or personalize.

4.2 Interfaces to External Systems or Devices

4.2.1 Software Interfaces

The system will use idp.ua.pt to authenticate the users, which is a identity provider form UA.

4.2.2 Hardware Interfaces

Besides the standard requirements of a computer, the system will need a NIC (network interface card) to be connected to the web, which is also a standard nowadays.

The system will also need a dns record, to be easy to remember it's address.

4.2.3 Communications Interfaces

Initially the site is only reachable from inside UA local network area or via VPN from the outside. In the future, it depends on the client desire.

5. Business Rules

5.1 <Rule class name>

5.1.1 <Rule name and ID>

6. System Constraints

Software implementation languages:

- Javascript.
- SQL

Third-party components or class libraries and versions:

- Javascript
- PostgreSQL
- NodeJS "0.8.x"
- NPM "1.1.x"
    - grunt
    - grunt-css
    - grunt-shell
    - grunt-compass
    - grunt-templater
    - grunt-coffee
    - growl
    - grunt-growl
    - express "3.0.2"
    - ejs
    - pg
    - sequelize "1.6.0-beta4"
    - gravatar
    - c3store
    - github
    - nodetime
    - async
    - passport
    - passport-github
    - dateformat

Operating system

- Linux.

7. System Compliance

7.1 Licensing Requirements

Since Node.js and PostgreSQL live under BSD-like licenses and the original project lived under a GPL-like license the team deliberated to ship the solution under a GPL License.
No non-free software will be needed which means there will be no need to buy or pay for any kind of software on which this solution will depend.

Information on GPL licensing:
http://www.gnu.org/licenses/gpl.html

Information on BSD licensing:
http://www.linfo.org/bsdlicense.html

7.2 Legal, Copyright, and Other Notices

Since this solution is the product of an academic project all new features will be intellectual property of University of Aveiro just like every other project born within this conditions.

7.3 Applicable Standards

So far there are no applicable standards to the solution.

8. System Documentation

Can be found on:
http://code.ua.pt/projects/uarobotfight/wiki