# CORSICA Prototype Product Description

# Onapha Rattanachottiteepakorn

CS411W

# Janet Brunelle

October 19, 2014

# Table of Contents

1 INTRODUCTION.............................................................................................................. 3

 1.1 Purpose .................................................................................................................. 4

 1.2 Scope ..................................................................................................................... 5

1.3 Definitions, Acronyms, and Abbreviations ........................................................... 7

1.4 References ............................................................................................................. 8

2 GENERAL DESCRIPTION............................................................................................. 10

2.1 Prototype Architecture Description ..................................................................... 10

2.2 Prototype Functional Description ........................................................................ 13

2.3 External Interfaces ................................................................................................. 3

List of Figures

Figure 1 - Major Functional Components Diagram ..................................................................... 12

List of Tables

Table 1 - Real World Product and Prototype Comparison Table .................................................. 6

Table 2 - Algorithms Description Table ...................................................................................... 13

1. **INTRODUCTION**

 When the amount of wages boosted by a bachelor’s degree prove to be worth the initial investment, it is no wonder that the amount of freshman admission increases over the year (Barrow and Rouse). The headcount statistic from Old Dominion University shows that from the year 2002 to 2013, the amount of first year students admitted increased from 2,481 to 4,550 per semester (Old Dominion University Student Headcount). According to an analysis by Pew Research Center, the household income from college graduated has been on the increase, helping with the nation’s economic growth (Fry). While there is certainly a rise in number of students entering college, it also presents a capacity challenge to universities across the nation. When a university failed to address the course capacity issue, the admitted students would be deprived of the opportunity to graduate in a 4 years timeline. This has led to the problem of the current class enrollment system becoming insufficient for handling course overflow.

 Old Dominion University currently relies on the Ellucian’s Enterprise Resource Planning, Banner, for handling the enrollment system. The school implemented Leo Online as the interface for accessing the information from Banner. While the current system provides a relatively stable services, it still underrepresents a way for students to get wait-listed into a full class online. This issue directly increases the amount of unnecessary communication between the students and the faculty. Going from students to professors to advisors and then to the enrollment staff, the long winded process of entering a full class proves to be an avoidable problem with an efficient online wait-list process.

(space left blank intentionally)

**1.1 Purpose**

 CORSICA is a an online wait-list management system. The product’s main features include the ability for students to enlist themselves into a wait-list of a full class and receive notification of the class’s availability. CORSICA also will also provide centralized services for Advisors, Professors, and as Schedulers. Such services include increasing class capacity, manual adding/dropping students in a wait-list, and manual opening/closing of certain classes.

 Student users will be able to log in to CORSICA with their ODU credentials. Once the user has been authenticate, the user will be able to search for a specific class and see its status. Once a full class becomes available, CORSICA will lock the course enrollment in order to prevent students not on the wait-list to register for the seat. The server would send out notification, through email and/or text message, that the class has become available to the users on the wait-list. The first user on the wait-list has 24 hours to register or refuse class before the seat is offered to the next student on the list. Once the user responds to the notification, he/she will be automatically removed from the wait-list.

 Advisors and Professors will also have access to CORSICA. In order for CORSICA to be a fully centralized automated wait-list system, faculty will have the ability to manually add or drop a student from the wait-list through the site. Scheduler will be a special type of user provided for the department’s class scheduler. The Scheduler will have the ability to manually close or open a wait-list for classes.

 CORSICA will also provide analysis for future class scheduling. The overall information will be only be accessible for the faculty users. This feature will display general statistics of students enrollment as well as providing prediction for course capacity in the proceeding semester. This tool should prove to be useful for schedulers to make sure that the university will be able to provide seats for its students equally.

 One of the major capabilities for CORSICA is the ability to handle classes that require labs and recitations. When a student enlists his/herself into a class that needs lab/recitation, CORSICA would alert the user to also list his/herself on to the desirable and available additional classes. Students will not be able to enroll into a class through CORSICA if he/she has not enlisted in the required classes respective wait-lists.

**1.2 Scope**

The prototype version of the envision product will be developed throughout the Fall 2014 semester at ODU. Since the time allotted to this project is only 18 weeks, CORSICA will not be able to ship as a real wait-list product to be used in real class enrollment process. The final prototype will simulate the way an online wait-list process operates while providing an example for a future wait-list process implementation.

 Since CORSICA does not have access to the Banner database, a simulation of students data will be created. CORSICA will receive CS classes information from the department’s scheduler in a form of JSON file. The data provides information on the classes’ overall status, such as total enrolled and capacity, as well as basic information, such as instructor information and the course reference number. With this information and the simulated students data, the prototype will demonstrate the key features of the envision wait-list system including login functionality, customized dashboard for each of the user types, enlisting in different wait-lists, enrolling into available class for student users, manual adding/dropping of students from wait-lists and manual closing/opening of wait-lists by faculty users, and providing a guide to use the website effectively for every users. CORSICA will also automatically send out notifications through email and text messages to the students to inform them of a class’s availability. Table 1 depicts the difference of the prototype and the envisioned product design.

|  |  |  |
| --- | --- | --- |
|   | **Real World Product** | **Prototype** |
| **Environments for all Users** | Yes | Will only demonstrate student, admin, and scheduler users |
| **Notification System** | Yes | Yes |
| **Check for available seats** | Yes | Yes |
| **Add Student to Wait-list** | Yes | Yes |
| **Drop Student from Wait-list** | Yes | Yes |
| **Mostly Automated** | Yes | Will rely heavily on user interaction |
| **Link to Banner** | Yes | Will be loaded with data.txt files instead |
| **Link to Leo-Online** | Yes | No |
| **Frontend Interface** | Yes | Yes |
| **Seat Analysis System** | Yes | No |

Table 1: Real World Product and Prototype Comparison Table

**1.3 Definitions, Acronyms, and Abbreviations**

**Algorithm**:A set of instructions that is carried out in order to solve a problem; typically used in the context of Computer Science.

**Banner**: The centralized academic and administrative records system used by Old Dominion University.

**C++**:An object-oriented computer programming language.

**CSS**:Cascading Style Sheets. A style sheet language used for formatting a document; typically used in conjunction with HTML for websites.

**Computer**: An electronic device that can store information and carry out programmed instructions.

**CORSICA**: Computer Science Class Waitlist. A piece of software developed for the Old Dominion University Computer Science department.

**Database**:A collection of (usually related) information.

**Email**: A method of sending messages from one computer to another on a network.

**GUI**: Graphical User Interface. A program that allows a user to interact with a computer by means of graphical icons such as buttons and text boxes.

**Heroku**: A cloud platform as a service supporting several programming languages.

**HTML**: Hypertext Markup Language. A markup language that is used for creating websites.

**Internet**: A worldwide system of interconnected computer networks.

**JavaScript**:A dynamic computer programming language, typically used in web browsers.

**Lab**:A section of a college class that commonly involves hands-on work.

**Lecture**: An oral presentation given by a college professor within a university setting.

**MySQL**:A relational database management system.

**Notification**:The act of informing someone of something.

**ODU**: Old Dominion University. A public post-secondary institution in Norfolk, Virginia.

**PHP**:A scripting language commonly used for web development.

**Prototype**: A preliminary model of something.

**Recitation**: A college class section where small groups of students from the corresponding lecture meet to review weekly material.

**Server**: A computer that provides information and services to other computers.

**SQL**:Structured Query Language.A programming language designed for managing data held in a relational database management system.

**Text Message**: An electronic message that is sent between cell phones.

**UIN**:University Identification Number. A unique identification number provided to students at Old Dominion University.

**Wait-list**: A list of people who are waiting for something.

**1.4 References**

"MySql vs MongoDB Performance Benchmark." *MoreDevs*. N.p., March 14, 2013. Web. Retrieved September 21, 2014 from http://www.moredevs.ro/mysql-vs-mongodb-performance-benchmark.

*Old Dominion University Student Headcount*. Rep. Old Dominion University, n.d. Web. September 21, 2014 from http://www.odu.edu/content/odu/about/facts-and-figures/factbook.htm[l](http://www.odu.edu/content/odu/about/facts-and-figures/factbook.html)

Barrow, Lisa, and Cecilia Elena Rouse. "Does College Still Pay?" *The Economists' Voice* 2.4 (2005): n. pag. Web. Retrieved September 21, 2014. http://transitions.s410.sureserver.com/wp-content/uploads/2011/09/barrow-rouse.pdf.

Fry, Richard. "The Growing Economic Clout of the College Educated." *Pew Research Center*. Pew Research Center, September 24, 2013. Web. Retrieved Septmeber 21, 2014. http://www.pewresearch.org/fact-tank/2013/09/24/the-growing-economic-clout-of-the-college-educated/.

Mardan, Azat. "PHP vs. Node.js." *Webapplog [programming Weblog]*. N.p., September 9, 2014. Web. Retrieved September 21, 2014 from http://webapplog.com/php-vs-node-js.

National Center for Education Statistics, U.S. Department of Education. (2001). Postsecondary Education. In *Digest of* education statistics 2001 (chap. 3). Retrieved September 6, 2014, from http://nces.ed.gov/programs/digest/d12/ch\_3.asp.

National Center for Education Statistics, U.S. Department of Education. (2001). Table 5: Number of education Institutions, by level and control of institution: Selected years, 1980-81 through 2010-11. In *Digest of* education statistics 2001 (chap. 3). Retrieved September 6, 2014, from http://nces.ed.gov/programs/digest/d12/tables/dt12\_005.asp.

National Center for Education Statistics, U.S. Department of Education. (2013). *Digest of Education Statistics: 2012*. Retrieved September 18, 2014.

Shaked, Uri. "AngularJS vs. Backbone.js vs. Ember.js - Choosing Your JS Framework." *Airpair*. N.p., August 21, 2014. Web. Retrieved September 21, 2014 from <http://www.airpair.com/js/javascript-framework-comparison>.

1. **GENERAL DESCRIPTION**

CORSICA is a part of Old Dominion University’s Computer Science Department’s professional workforce development course. The creation of the prototype of a new online wait-list process will provide stimulate an implementation within the Computer Science classes. The prototype will allow student users to login to the CORSICA website with their ODU credentials. The user will be able to enlist his/herself to a full CS class. Once the class becomes available to enroll, CORSICA will automatically send a notification to the enlisted students through email and/or text message. While the prototype will not directly communicate with ODU’s database, it will serve as a good proof of concept for an online wait-list process implementation.

**2.1 Prototype Architecture Description**

CORSICA’s main structure includes backend server using NoSQL database MongoDB with Node.JS as the tool to communicate between the backend and frontend. MongoDB uses the NoSQL database structure which accommodates data from the already existed CS classes information in JSON format (Mardan). CORSICA will utilize Backbone.JS as its main scaffold for the site’s structure. Backbone.JS “gives structure to web applications by providing models with key-value binding and custom events, collections with a rich [Application Programming Interface (API)] of enumerable functions, views with declarative event handling, and connects it all to [the project] existing API over a RESTful JSON interface” (Backbone.js).

CORSICA will receive the current semester’s CS classes from the scheduler’s JSON file. The file will provide necessary information to be input into the class’s database via javascript code. The simulated users data will also be stored manually into the users database for testing purposes. The data will provide different types of user including: an administrator, advisor users, a scheduler user, and student users. Once all of the data has been stored into the databases, CORSICA will allow for interaction via a web app interface created by the Backbone.JS framework. The development for the frontend of the websites will be utilizing the basic knowledge of HTML, SASS (Ruby based pre-processor for CSS), Javascript and jQuery (a client-side Javascript library). Figure 1 illustrates the major functional components for the envision product.

(space left blank intentionally)



Figure 1: Major Functional Components Diagram

**2.2 Prototype Functional Description**

For CORSICA to be fully functional for the final prototype shipment, the application will need at least eight majors algorithms. Table 2 illustrates the full usage of each of the algorithms to be included in CORSICA prototype.

|  |  |
| --- | --- |
| **Algorithm** | **How it functions** |
| **Load Enrollment Data Files** | * Course data files are loaded into CORSICA.
* Files contain class’s capacity, number of enrollments, and available seats.
 |
| **Open Course** | * An administrator or scheduler user logs into CORSICA and opens a course for students to enroll in.
 |
| **Check for Open Seats** | * Once a course becomes full, a wait-list is activated for it by CORSICA
* CORSICA will continually reference the current course capacity and amount of students enrolled.
* If the amount of students enrolled is less than course capacity, a seat has become available.
* Calls notification algorithm
 |
| **Add Student to Wait-list** | * Student X wishes to enroll in Course Y’s wait-list
* CORSICA receives this request and adds Students X to wait-list queue
* Course Y’s wait-list is updated
 |
| **Notification** | * The check for open seats algorithm completes and returns true for an available seat
* All students on the wait-list queue are notified of opening
* Students respond
 |
| **Drop Student from Wait-list** | * Student X wishes to be dropped from Course Y’s wait-list or the time window for that student has expired
* CORSICA receives this request and removes Student X from the wait-list queue
* Course Y’s wait-list is updated
 |
| **Increase Course Capacity** | * Administrator logs into CORSICA and increases course capacity for Course Y
 |
| **Close Course** | * An Administrator or Scheduler user logs into CORSICA and closes the wait-list for the course
 |

Table 2: Algorithms Description Table

The user roles in the prototype will only include an administrator, advisor users, a scheduler user, and student users. While the initial product design aims to automatically grab the user authentication through the university’s MIDAS ID (ODU’s login and password management system), the prototype will be getting the users data through manual insertion.

Once a user is logged in, CORSICA will display a specific dashboard customized the the specific user’s role. The administrator, advisor, and scheduler will be able to see the current semester’s wait-list, both active and inactive, as well as having the ability to click through for a detailed view of a specific wait-list. The faculty users will have the ability to customize the wait-list in their own way. Advisor will be able to move, add, or drop a student into a wait-list, while scheduler will able to completely close the wait-list or open a new one. The administrator will have the access to both advisor’s and scheduler’s abilities.

The dashboard for a student user will only display the user’s enlisted wait-list. From the dashboard, the user will be able to search for a specific class through class’s name, instructors, or the course reference number. If the wait-list is available (opened and not full), the student will be able to click through to the class’s detailed page, which will display the number of students currently in the list while providing basic information about the class. The student may join the wait-list if he/she met the requirement as well as opt for a notification of available seat through email and/or text message. CORSICA will automatically email the users on the list of the seat’s availability and give the first user a 24 hours period to respond. The user will be removed from the list once responded or if the 24 hours have passed since the sending of the notification.

CORSICA’s prototype will also give the ability for a nonessential personnel such as parents or any interested party to read about the project. The static pages will be created for people without the ability to log in to see CORSICA’s product description, FAQ, and the developers’ about page.

One of the major capabilities that will be implemented into the prototype product is the ability for CORSICA to link wait-lists of requisite classes such as the lecture, recitation, and lab requirement for many of the programming classes. As the information will be provided by the scheduler’s classes data, CORSICA will only have to connect the classes together and create restriction rule for students wait-list enlisting process.

**2.3 Extended Interfaces**

CORSICA prototype does not rely on any third party interfaces. While the Team Webpage is hosted on CS server, the current prototype is hosted on Heroku with Mongo Cloud server for the database. The plan is to eventually move the prototype to the CS server and completely remove the links to Heroku production site.