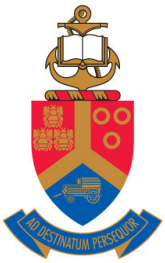

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

COS301 Mini Project

Architectural Requirements Specification

Group 5a

Matthew Gouws *u11008602*
Tsepo Ntsaba *u10668544*
Werner Mostert *u13019695*
Semaka Malapane *u13081129*
Andrew Parkes *u12189139*
Name Surname *uXXXXXXXX*
Name Surname *uXXXXXXXX*
Name Surname *uXXXXXXXX*

Version
March 9, 2015

Contents

1 Introduction

The Buzz system is to be developed to enhance user interaction by means of a online discussion board, this board will have multiple components to allow for better user interactions by means of rewards, ranks and privileges. This document serves as the Architectural specifications for the Buzz system. Including the constraints imposed by the client Ms. Vrede Pieterse.

2 Vision

This project should help to engage students and encourage learning amongst students. The system should be able to be integrated into any website to further help other universities with the same results.

3 Background

- The system is to be developed to help the CSEDAR (Computer Science Education Didactic and Applications Research.
- Improve on the current discussion board already in place for the University of Pretoria Department of computer science.
- Create a collaborative group of students to help solve problems before escalating to a higher knowledge source.

4 Architectural Requirements

4.1 Access channel and integration requirements

4.1.1 Access channels

Human access channels

System access channels

4.1.2 Integration Channels

Integration Channel

- The Enterprise service bus will be used. The bus is a subsystem that facilitates the communication between other subsystems. This means that subsystems only need to communicate through the ESB. This significantly lowers the amount of necessary connections and it processes the messages, meaning other subsystems need not waste time processing messages. If a subsystem is replaced or added, the only changes that need to take place is the creation of a new interface between the new system and the bus. This

makes the system much more flexible and makes communication much more simple.

Protocols Used

- Hypertext Transfer Protocol

  HTTP is the standard way and method to transfer method of web pages over the internet by using the protocols TCP/IP (Transmission Control Protocol and Internet Protocol) to manage the Web transmission. In use of Buzz all posting, registering and even just access to the forums will make use of these protocols.

- Mail Protocols POP3 and SMTP

  Post Office Protocol (POP) will probably not be used within the system but is used in receiving emails. Simple Mail Transfer Protocol (SMTP) will be used though, the protocol is used in the sending of emails by making use of a SMTP server. In the case of Buzz all notifications that are not present within Buzz will be sent through email such as account verification and notification of changes within a users account. Finally if a user loses account details all information and lost password would be sent over email.

- File Transfer Protocol

  File Transfer Protocol (FTP) is the method of copying files over the internet or a network but more simply put allows for the transfer of information from one computer to a remote computer. In terms of Buzz we are not creating a File Server but rather with all the images profile pictures and any other documentation that are uploaded to different profiles for other users to use make use of the File Transfer Protocol.

4.2 Architectural responsibilities

- The System must be able to provide concurrent clients to read threads, post threads and update threads.
- The system should be able to store all threads and posts, as well as who posted, and whether a thread has been deleted.
- The system should provide an integration environment to allow for multiple deployment
- The system could allow for persistent data storage for easy 'Remember Me'
- Storage of archived thread

4.3 Quality requirements

4.3.1 Scalability

The Buzz system , when neccesary or needed must be able to be easily converted into an application to be used on mobile phones. The system should be built using independent components (MVC pattern) to ensure separition of concerns,cohesion,decoupling of components and pluggability when developing it for mobile.

4.3.2 Performance requirements

The system needs to respond almost instantly in terms of creation and deletion of a Buzz space or creation and deletion of threads as well as the management or administration of the whole system and it must also be able to handle high contention where a lot of concurrent users will want to access an actice buzz space at once where they should not experience latency or be susceptible to data traffic.

4.3.3 Maintainabilty

The system should be able to be continually improved by developers who worked on it initially and new developers.The system should also have the attribute that it can be restored.

4.3.4 Reliability and Availability

The buzz system needs to be 'online' almost every time.The server or host for the system should then be able to be used by users at any time. Also , threads or post must be recent and for the administrators the system must produce correct results when queried.

4.3.5 Security

The system as a whole should be able to protect the information of students and lectures from external infiltration e.g possible SQL injection. Again, it should provide some form of validation mechanism to ensure that only students that are allowed on specific Buzz spaces are allowed on a certain one. Also , the mechanism needs to block out all students who are not allowed on the system.

4.3.6 Monitorability and Auditability

Buzz should provide a way such that it can be audited or viewed at particular check points i.e it can keep track of the number of posts, keep track of dates and be able to link a post to a student. This will require to keep a record of its states during different days or months and with this we can employ the Memento design pattern.

4.3.7 Maintainability

The system should be able to be continually improved by developers who worked on it initially and new developers. The system should also have the attribute that it can be restored.

4.3.8 Usability

The system must be user friendly and easy to use for users. Users must be able to interact with the system i.e the user interface must be easy to learn and understand.

4.3.9 Integrability

It should be easy to combine The Buzz system with other modules or assistance technologies like a Database Management System.It should be able to export data into various other modules.

4.4 Architecture constraints

Reference Architecture

- Java EE (Enterprise Edition) is specified as the chosen Reference Architecture to use.

  Details

  Java Platform, Enterprise Edition or Java EE is Oracle's enterprise Java computing platform. The platform provides an API and runtime environment for developing and running enterprise software, including network and web services, and other large-scale, multi-tiered, scalable, reliable, and secure network applications. Java EE extends the Java Platform, Standard Edition (Java SE), providing an API for object-relational mapping, distributed and multi-tier architectures, and web services.

  Comments

Other Software Technologies

- JPA (Java Persistence API)

  Details

  The Java Persistence API (JPA) is a Java programming language application programming interface specification that describes the management of relational data in applications using Java Platform, Standard Edition and Java Platform, Enterprise Edition.

  Comments

- JPQL (Java Persistence Query Language)

  Details

  The Java Persistence Query Language (JPQL) is a platform-independent object-oriented query language defined as part of the Java Persistence API (JPA) specification. JPQL is used to make queries against entities stored in a relational database. It is heavily inspired by SQL, and its queries resemble SQL queries in syntax, but operate against JPA entity objects rather than directly with database tables.

  Comments

- JSF (JavaServer Faces)

  Details

  JavaServer Faces (JSF) is a Java specification for building component-based user interfaces for web applications and exposing them as

server side Polyfills.] It was formalized as a standard through the Java Community Process and is part of the Java Platform, Enterprise Edition.

Comments

- HTML (HyperText Markup Language)

  Details

  HyperText Markup Language, commonly referred to as HTML, is the standard markup language used to create web pages.

  Comments

- AJAX (Asynchronous JavaScript and XML)

  Details

  Ajax is a group of interrelated Web development techniques used on the client-side to create asynchronous Web applications. With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page.

  Comments

- CSS

  Details

  Cascading Style Sheet will be used for the formatting and styling of the discussion forum. It will ensure separation of the content from presentation.

  Comments

- Github

  Details

  Github will be the platform used by the developers to collaborate. This will be where all the code will be pushed so that the developers can be able to work together from remote locations and be able to combine all their code/work.

  Comments

Operating System

- Linux

  Details

  The forum will be designed to work on any Operating system. However, Linux is ideal for certain functionalities therefore the forum will work optimally on Linux Operating Systems. The

web server will be hosted on a Linux machine as Linux is widely considered to be the best operating system for web servers.

Comments

Deployed Environments

- Web Based Interface

  Details

  Users will be able to navigate to the application using a web browser from any compatible device, such as a tablet; smartphone or computer.

  Comments

  This is a very suitable environment since it allows for a large measure of ease of access.

- Android (out of scope - for future reference)