**G52GRP**

**Democratic Conferencing Tool**

**Interim Report Appendix – November 2009**

* **To Accompany Main Report -**

**Group:** gp09-sdb

**Supervisor:** Dr. Steven Benford

**Group Members:** Robert Golding, William Redrup**,** Tammie Seo**,** Christopher Lensvelt**,** Henry James**,** Zhongda Zhu

# Market Research - Appendix

## Bantu

<http://www.bantu.com/>

Bantu is a cross-platform Business IM, and is similar to Google wave, but less user-friendly.

## WeMeeting

<http://www.netdive.com/indexwme.htm>

WeMeeting is another Business IM which incorporates video-conferencing into the rooms.

## ZPN Mesh

<http://www.zullotech.com/>

ZPN Mesh is a project collaboration package, with a built-in IM system.

# Technical Research - Appendix

## Django

Here is an example of a simple model in Django source code:

class Poll(models.Model):

question = models.CharField(max\_length=200)

pub\_date = models.DateTimeField('Date published')

class Choice(models.Model):

poll = models.ForeignKey(Poll)

choice = models.CharField(max\_length=200)

votes = models.IntegerField()

Here’s an example of a simple view in Django:

def index(request):

latest\_poll\_list = Poll.objects.all().order\_by('-pub\_date')[:5]

data = {'latest\_poll\_list': latest\_poll\_list}

return render\_to\_response('index.html', data)

Here is an example of a template in Django:

{% if latest\_poll\_list %}

<ul>

{% for poll in latest\_poll\_list %}

<li>{{ poll.question }}</li>

{% endfor %}

</ul>

{% else %}

<p>No polls are available.</p>

{% endif %}

## Ruby-on-Rails

To create the blog application:

$ rails blog

Here is an example code to configure MySQL Database in Ruby-on-Rails:

development: adapter: mysql encoding:

utf8 database: blog\_development pool: 5 username: root password: socket:

/tmp/mysql.sock

An example of Migration in Ruby-on-Rails:

class CreatePosts <

ActiveRecord::Migration def self.up create\_table :posts do |t| t.string :name

t.string :title t.text :content t.timestamps end end def self.down drop\_table

:posts end end

## SVN

Here is an example displaying the basic structure of the SVN 3D-Tree system.



## Mercurial

The hashes in a Mercurial repository look like this:

1:1ef7872431f9c64908c732f0bcd4db5700b4cb70

This means that unlike Subversion, two revision identifiers cannot be easily compared to see which is newer.

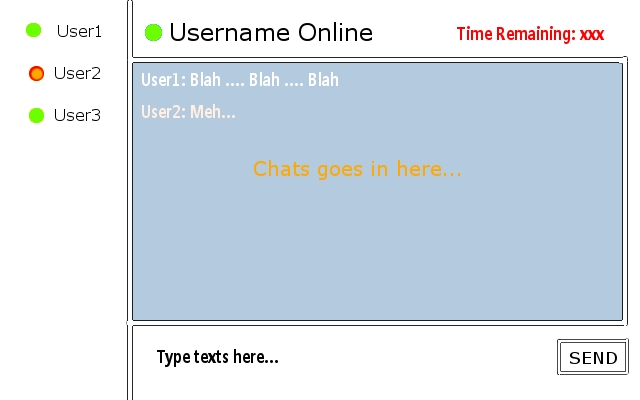
# Functional Specification – Appendix

## User Interface

Initial Design #1



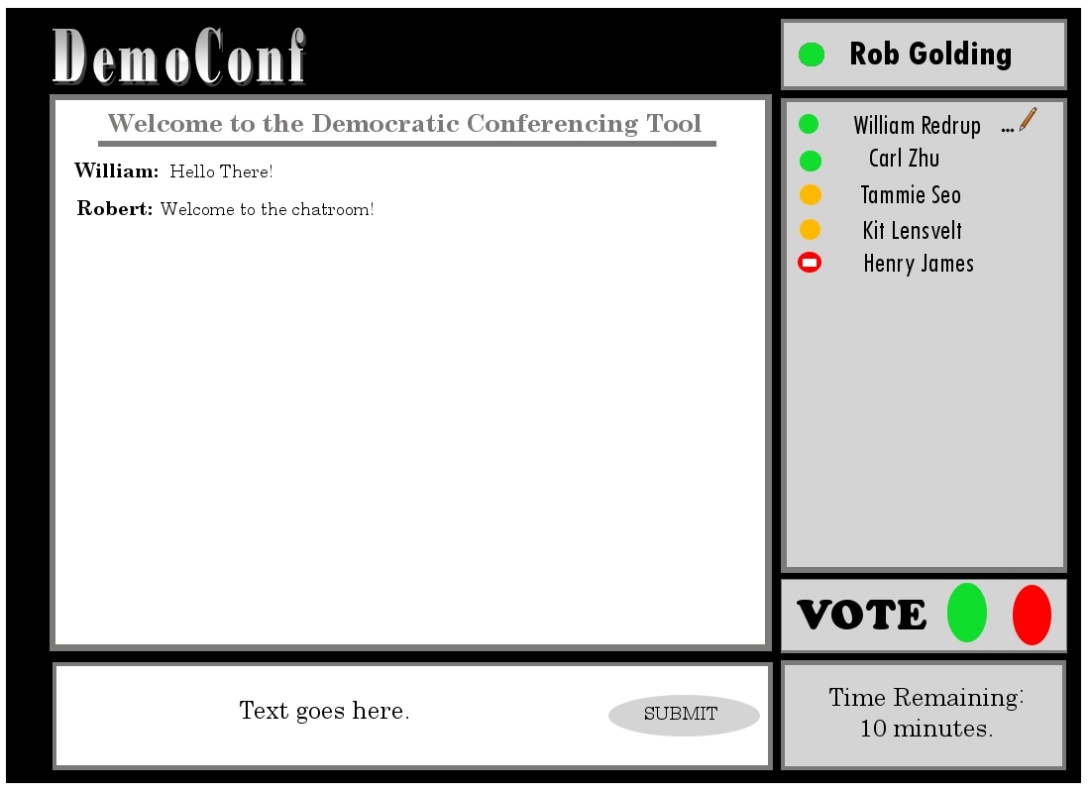
Initial Design #2



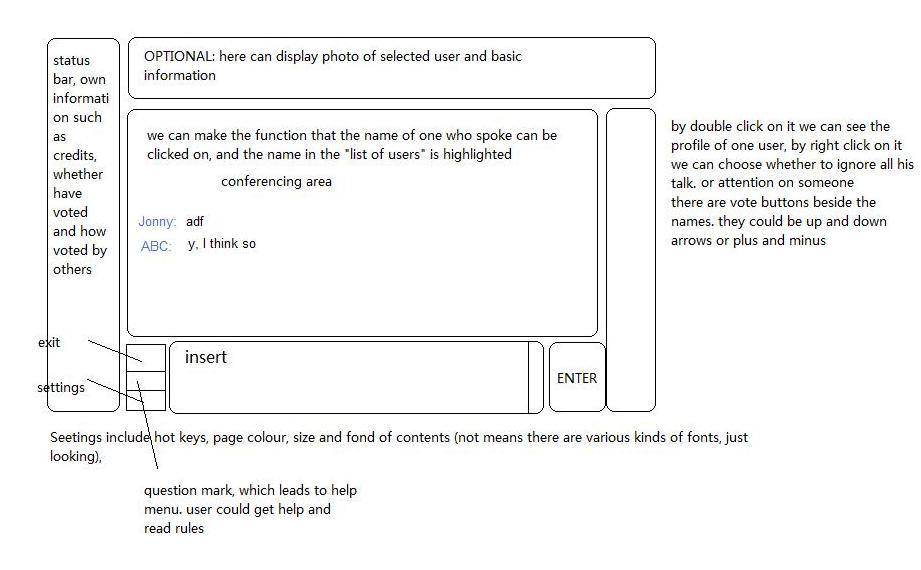
Initial Design #3



Initial Design #4



Initial Design #5



# Prototyping

Here are some screenshots of the system prototype, to show the initial workflow.

*Note: The lines bordering each image are purely there to help identify and separate each picture. The actual prototype does not have this feature.*

Basic Room List:



Log-in Fields:

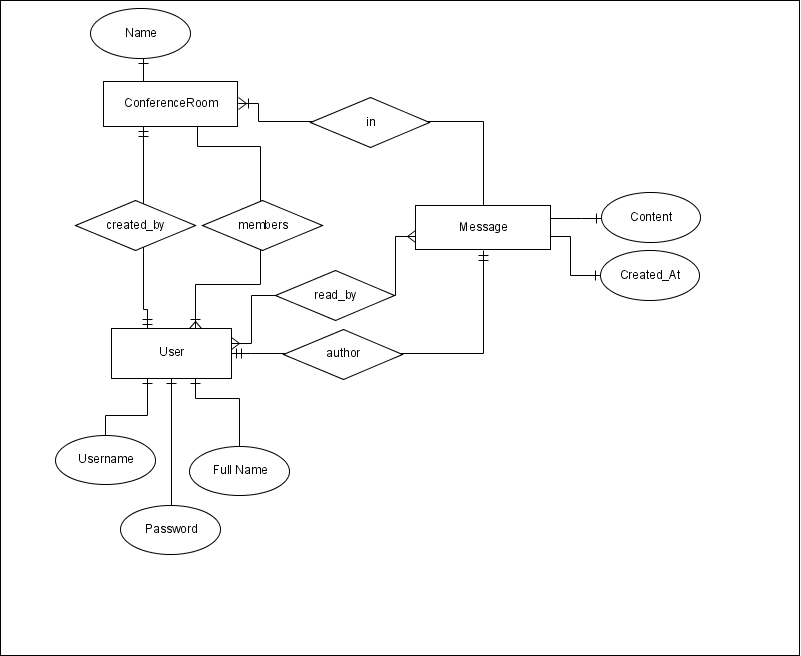


Basic Room Example:



# Implementation Decisions – Appendix

## Entity-Relationship Diagram



## Models.py

class Room(models.Model):

name = models.CharField(max\_length=255)

current\_members = models.ManyToManyField(User, editable=False)

def get\_and\_mark(self, user):

messages = self.messages.exclude(read\_by=user)

for message in messages:

message.mark\_for(user)

return messages

def get\_unread(self, user):

return self.messages.exclude(read\_by=user)

def get\_absolute\_url(self):

return reverse("conference\_room", args=[self.id])

def \_\_unicode\_\_(self):

return self.name

class Message(models.Model):

room = models.ForeignKey(Room, related\_name="messages")

author = models.ForeignKey(User, related\_name="messages")

content = models.TextField()

read\_by = models.ManyToManyField(User, blank=True, related\_name="read\_messages")

created = models.DateTimeField(auto\_now\_add=True)

def mark\_for(self, user):

self.read\_by.add(user)

def \_\_unicode\_\_(self):

return "[%s] %s" % (self.room.name, self.content)

## Low-Tech Prototyping

### Photographs

### Rule-Sets

#### Set 1

* Users have a hard limit of x words per interval.
* The word count is reset every interval.
* Donations are not allowed.
* Users can vote to silence someone for the next x intervals.
* Conferences have a “chair” to lead the session.

#### Set 2

Same as **Set 1**, but:

* Donations are allowed from one user to another.

#### Set 3

Same as **Set 1**, but:

* Users can also vote to increase someone’s word limit. No-one loses any words from their limit.

#### Set 4

* Conference is similar to “pass-the-baton”, where only one (or two) person is/are allowed to talk at a time.
* The baton would be passed on after a number of words are expended.
* Users can request more words if they are running low.

#### Set 5

Same as **Set 4**, but:

* Users can vote the current speaker “up” or “down”, to either increase or reduce their word allowance – if they agree with what they are saying.

#### Set 6

* The meeting chair is not allowed to speak. Instead the chair’s responsibility is to moderate the session – deciding whether to overrule decisions.
* This could be decided by a vote, or selected when the meeting is created.

#### Set 7

* The chair can speak, with an unlimited budget – and their text is highlighted to make it stand out.
* Users can flag the chair for getting involved in discussions where it is not allowed.
* A new chair could possibly be chosen if too many flags are raised.