Smart Neighbourhood

Smart Neighbourhood® (SN for short) is a web-based portal for communities to manage their daily activities efficiently. There are several aspects to SN which makes it an indispensible element for communities.

The SN knowledge base comprises of “gyan” contributed by the community members. These may include knowledge about contact details of plumbers, electricians, how to pay property tax, etc. The set of things about which knowledge is stored, is decided by the community members at run time. Knowledge elements can be rated by others based on how useful they found it.

The SN decision center comprises of collective decisions taken by the community on various issues. An issue is usually put forth by a community member at the end of which there are one or more questions on which a vote is taken. The choices on which to vote, is decided by the member posting the issue. Other members may comment and vote. Some voting systems may require anonymous voting. No member may vote more than once. Some voting systems may allow interim results to be shown, while others may want to block interim results until the voting is complete. Some voting systems may allow anyone to vote, while others may require voters to possess certain credentials (like office-bearers of the community association) for them to vote.

Any user, after logging in, should get a quick overview of what is happening in the community as part of his/her timeline. Other features like connecting with google maps or displaying relevant news feeds etc. will make the platform much better.

Deliverables

Mandate – 1:

Create a canonical systems requirements specification from the above proposal by explicating assumptions and making the proposal complete. Classify requirements into analysis and design classes and in turn classify the design class into required (obligated), optional (permitted) and forbidden characteristics.

Create a UML system model as a high level design specification for the intended system. Calibrate your model based on metrics like Essentialism, Symmetry, Cognitive load and Universality (DRY principle)

Mandate – 2:

Create a workflow model (declarative modeling of system dynamics) for your design specification, detailing essential and optional workflows, their states, specializations of workflows if any and constraints within and across workflows.

Create a low-level schema for your model detailing tables, columns, key constraints, referential integrity and index structures. Specify your model in the form of a Rails migration and create a scaffold for your application.

Mandate – 3:

Implement the workflows detailed above as controller actions and provide interfaces to them through necessary views. Augment your application with third party libraries as applicable (jQuery, Oauth, etc.)

Run Unit tests on your application to verify workflows and model class behaviours

Mandate – 4:
Enhance your application to support multiple simultaneous users. Optimize your application for scalability, performance, resilience and crash recovery. Use a mongrel load balancer to distribute interaction load over several servers.

Any file system or physical database tuning for improving performance is an added bonus.