

Software Requirements Specifications

Preliminary Design

The Center for Socially Relevant
Computing Website

Team 47

Contents

Topic	Page #
I. Introduction	4
i. Purpose	4
ii. Definitions	4
iii. System Overview	5
II. Overall Description	6
i. Functional Requirements	6
User Interface	6
Figure 1 (Top Level Diagram)	7
Figure 2 User Interface	8
Web Backend	9
Figure 3 Web Backend	9
Databases	10
Figure 4 Databases	10
Payment Gateway	11
Figure 5 Payment Gateway	11
ii. User Screens	12 - 21
iii. Dependencies, Constraints and Limitations	22
III. Change Management	23
IV. Cross Reference Listing	24
V. Integration Thread	25

I. Introduction

i. Purpose

The CSRC needs a website to promote its good works. It needs to be able to bring together students and teachers from various universities and help form a nation-wide community. It has to facilitate project design management and implementation among the community members and should enable them to solicit supporters and investors.

A dynamic website serving the cause and at the same time being simplistic and easily navigable needs to reflect what the CSRC is and needs to clearly outline its mission and its goals. A set of pages, which contain the features and functionality as mentioned in the functional requirements would have to be designed and built.

This document is intended to outline the various requirements and is a complete description of the behavior of the system to be developed. It includes the Software Architectural Block Diagrams, which indicate the major software modules and how communication happens between these modules. It also shows the various data storage destinations.

Further we list down the various dependencies, assumptions and constraints of the system that will be developed. We also provide a change management form, which will be used in cases when the customer requests a requirement change.

ii. Definitions

- **SRS** – Software Requirements Specification
- **Django** – High level web development framework, built with Python programming language. It provides an Admin Interface, CMS customizability and ORM functionality.
- **CMS** – Content Management System
- **ORM** – Object Relational Mapping – Data management using techniques which allow data convertibility and inherent CRUD (Create, Read, Update, Delete) functionality.
- **SRC** – Socially Relevant Computing
- **CSRC** – Center for Socially Relevant Computing
- **HTTP** – Hypertext Transfer Protocol
- **AJAX** – Asynchronous JavaScript and XML – Web development technologies used on the client side for creating asynchronous web applications.
- **mod_wsgi** – A web server module which provides an interface for hosting Python based web applications.
- **API** – Application Programming Interface

iii. **System Overview**

The software system to be developed for the CSRC website helps manage software projects and brings together a community of like-minded people working towards a single goal.

The users of this system could be students working on various projects and/or those who are interesting in working on new projects and in taking new initiatives. It provides them various pages like “Get Involved”, “Projects”, “Publications”, “Gallery”, “Resources” which will create interest in the community and help spread the word through sharing options.

The admin interface provides administrators of the website with options of adding a new page, editing existing pages, deleting existing pages. It also provides the admin with the option of adding, modifying and deleting projects from the database.

The major modules in the system are the User Interface, the Web Backend, Databases and the external Payment Gateway, which will be integrated using third-party API.

II. Overall Description

i. Functional Requirements

As illustrated below in Fig 1, the major functional software modules are:

- User Interface
- Web Backend
- Databases
- Third Party Applications/Tools

User Interface

For any visitor to the CSRC website, the user interface is the collection of all pages which could be shown. These pages as mentioned in Figure 2 could be categorized into the following:

- Static Pages
Pages like the About CSRC page, Press/Media/Publications pages, Gallery Page, Resources Page and the Thank You Page can be termed as Static Pages. These pages can be maintained using the CMS functionality provided by Django.
- Dynamic Pages
Pages like the Get Involved page, Projects Page, Login Page and the Contact CSRC page would be developed and maintained using the Django views and controllers which provides options for integrating the HTML templates with Django forms and model objects.
- Special Pages
These are specific pages like the 404 Page Not Found page, 500 Internal Server Error Page, 503 Service Unavailable page etc. All pages, which return specific HTTP Status Codes, would be categorized under this.

Admin Pages

The administrator of the website has special privileges. She/he can login to the admin interface using a LOGIN Page. Once logged in they will be taken to the admin page which will in turn provide them options to add/edit/delete any pages to the website. They will also be able to view, add, modify, delete projects from the database.

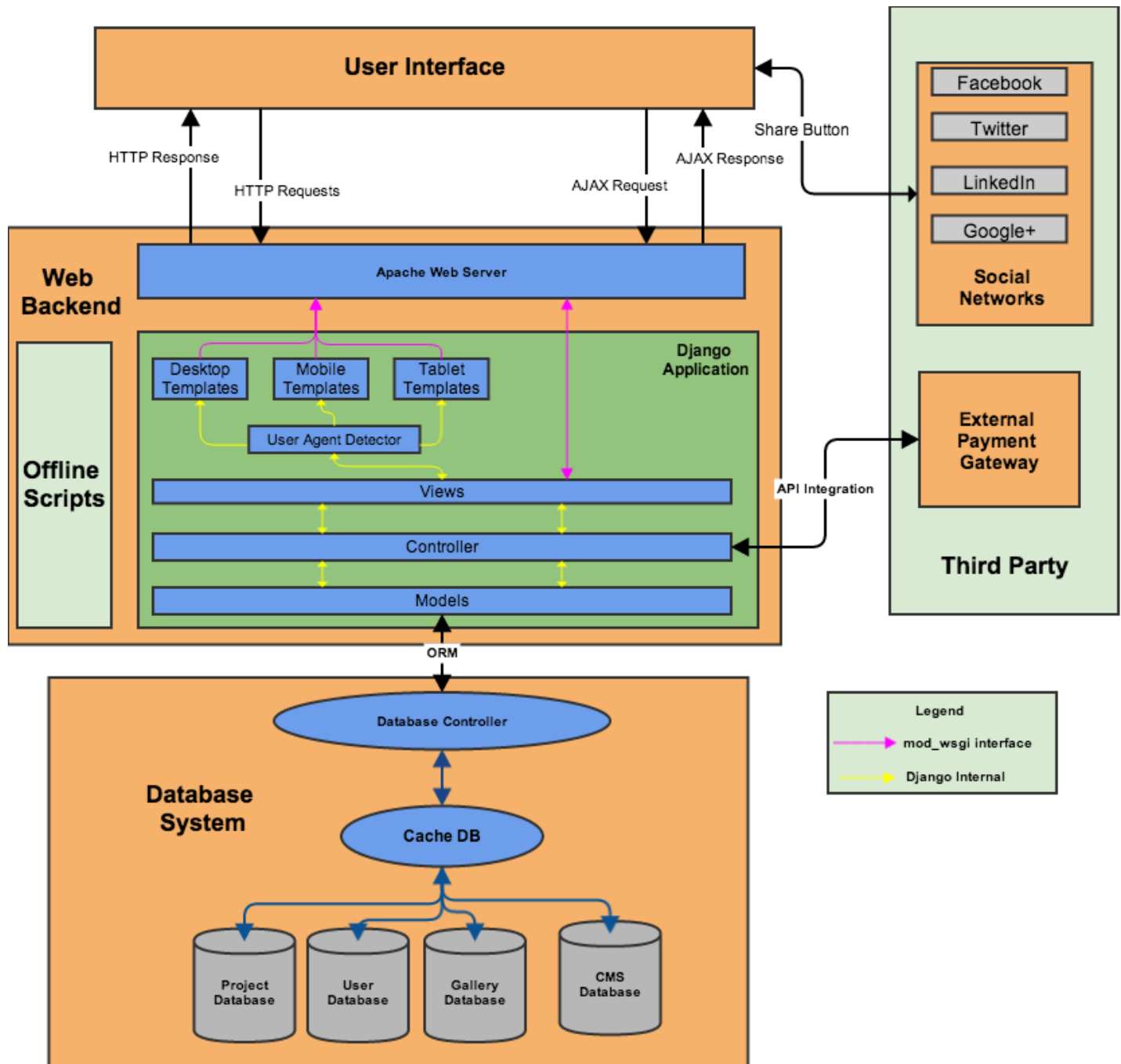


Figure 1 - Top Level Diagram

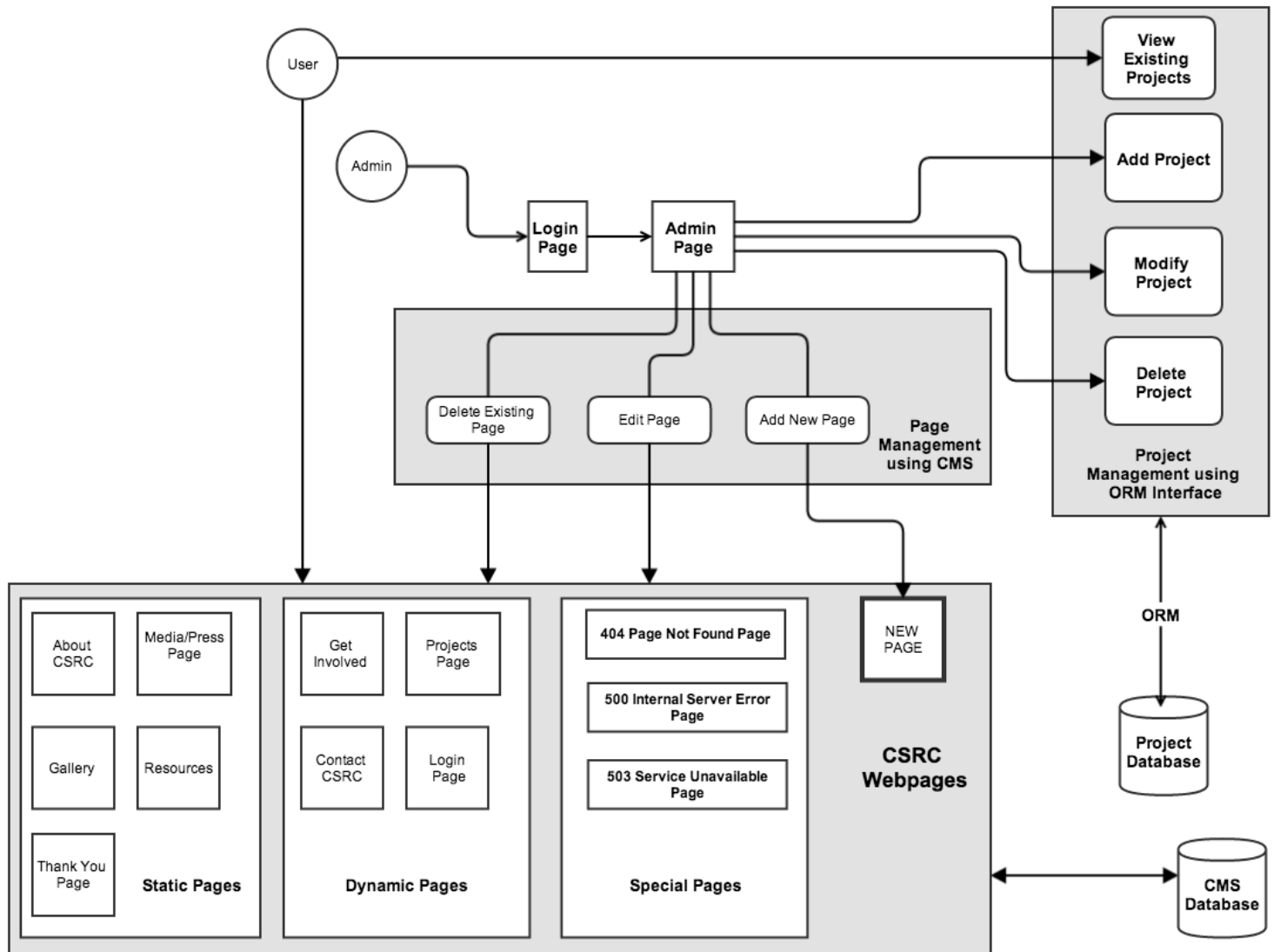


Figure 2 - User Interface Diagram

Web Backend

This includes the various software that is present in the web server. These include the entire LAMP Stack, viz,

- Linux,
- Apache,
- MySQL,
- Python.
- The web application framework used is Django and a web application is created using that.
- It is integrated with the Apache HTTP Web server
- Integration between web server and Django application is done using mod_wsgi which is an HTTP Server module for integrating Python based applications.
- The Django application is further subdivided into Models, Views, Controller, and Templates.
 - o **Models** – For integrating with the database (MySQL) using the Django ORM.
 - o **Views** – These are the part of the code which decide which templates are to be rendered for what URLs.
 - o **Controller** – This part of the Django application contain code that integrate the views with the models. These also contain code to integrate with third party tools and applications like the External Payment Gateway.
 - o **User Agent Detector** – This is a middleware which sniffs the user agent received in the request and decides which kind of template is to be returned.
 - o **Templates** – These are Django templates which are built using Django forms and HTML/CSS which are in turn rendered in the user's Web Browser.
- The server backend also contains **Offline Scripts** for sending notifications to users. This also sends the welcome mails to users.

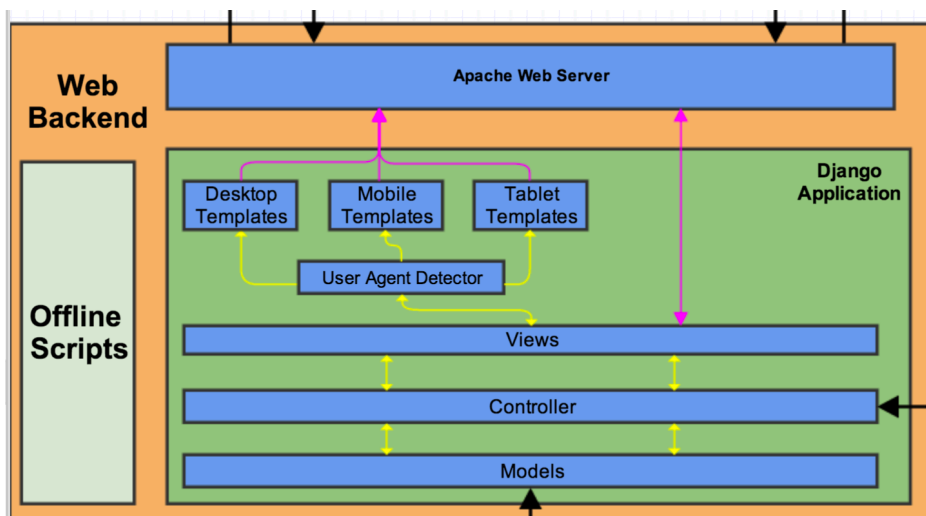


Figure 3 - Web Backend

Databases

Various types of databases are used to store the relevant information generated using this application. They are listed as follows:

- Project Database
They contain information about the various student projects that are/were listed on the website. They could be current projects, completed projects, suggested projects etc.
- User Database
This contains information about the various backers who had signed up for email notifications. This database also contains information about the various students, mentors, faculty members and community participants.
- Gallery Database
This database is a document-oriented database that contains images and videos related to various projects.
It also stores images related to the generic pages.
- CMS Database
This database stores all the content that is shown on various pages.
These can be modified through the admin interface using CMS.

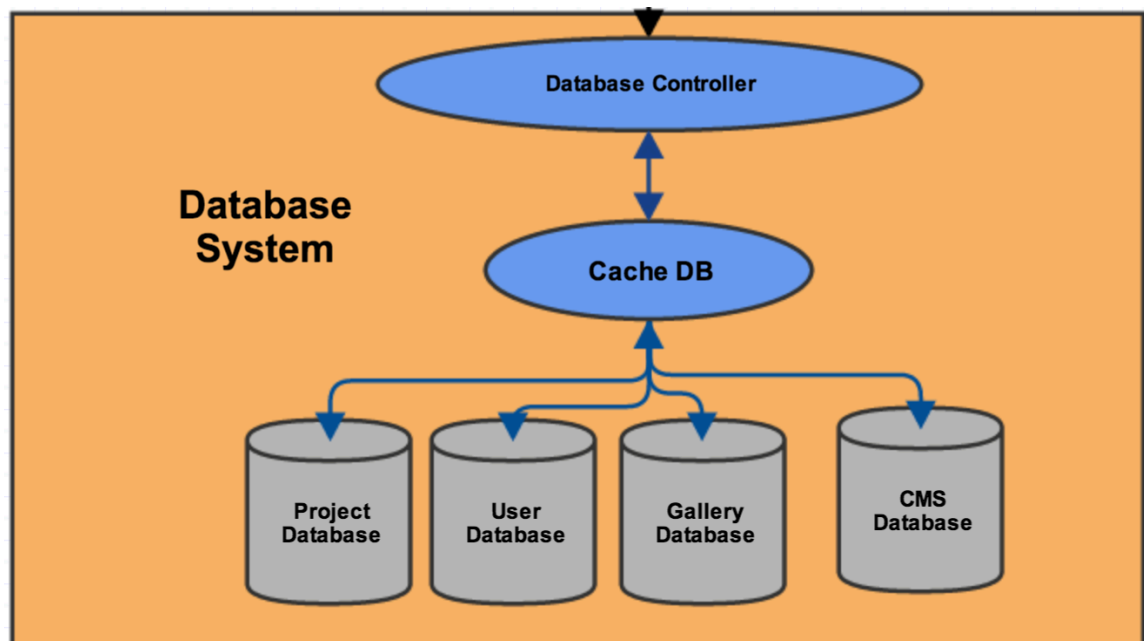


Figure 4 - Databases

Payment Gateway

- The website's Project Page and the Get Involved pages contains links for visitors to become backers of the project.
- On clicking the link they will be taken to a page where the project description is shown and the user is asked to enter some preliminary information.
- Once the user enters these values, we make a request to the Payment Gateway (PG) API and when successful, redirect the customer to a **secure** page on the PG website.
- The user is prompted to enter credit/debit card details.
- The user is prompted to enter billing information.
- Once the user confirms, the money is withdrawn from his account and the user is then redirected to the CSRC website's Thank You Page.

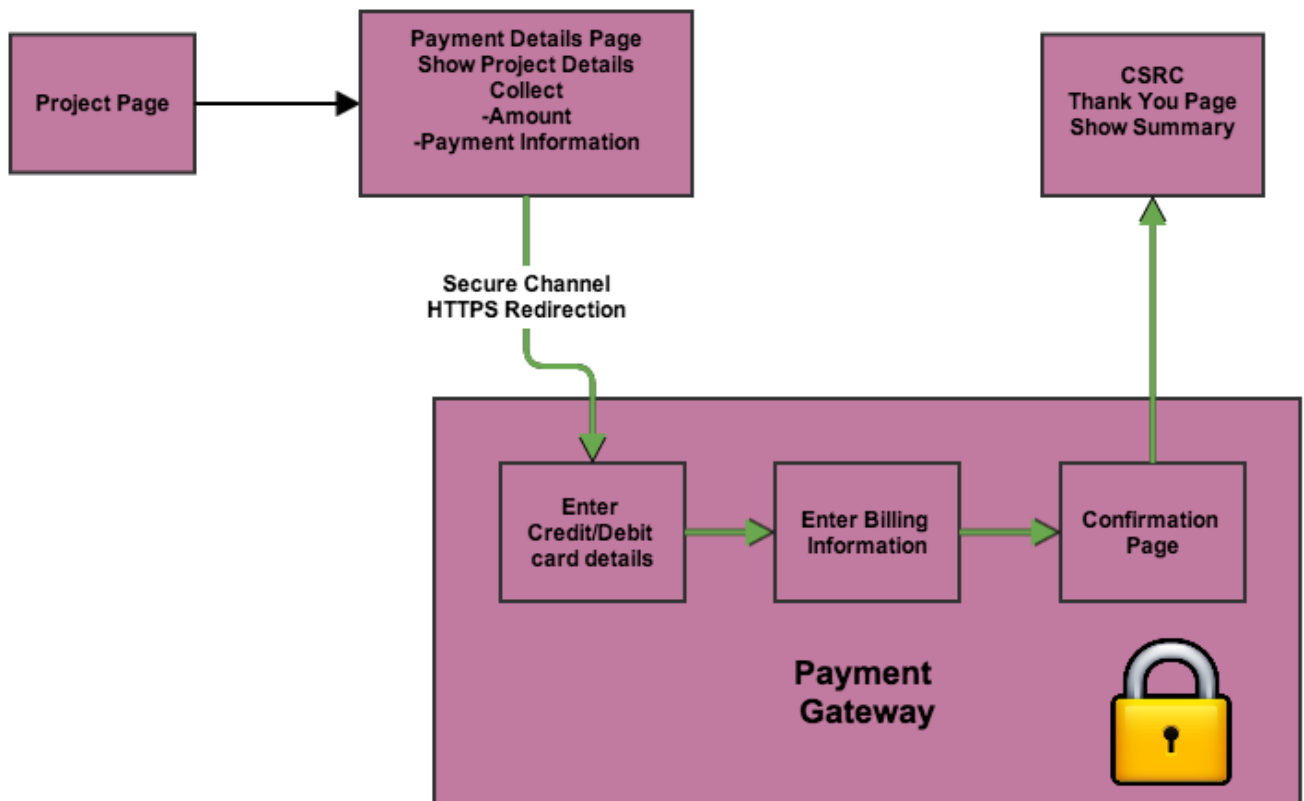
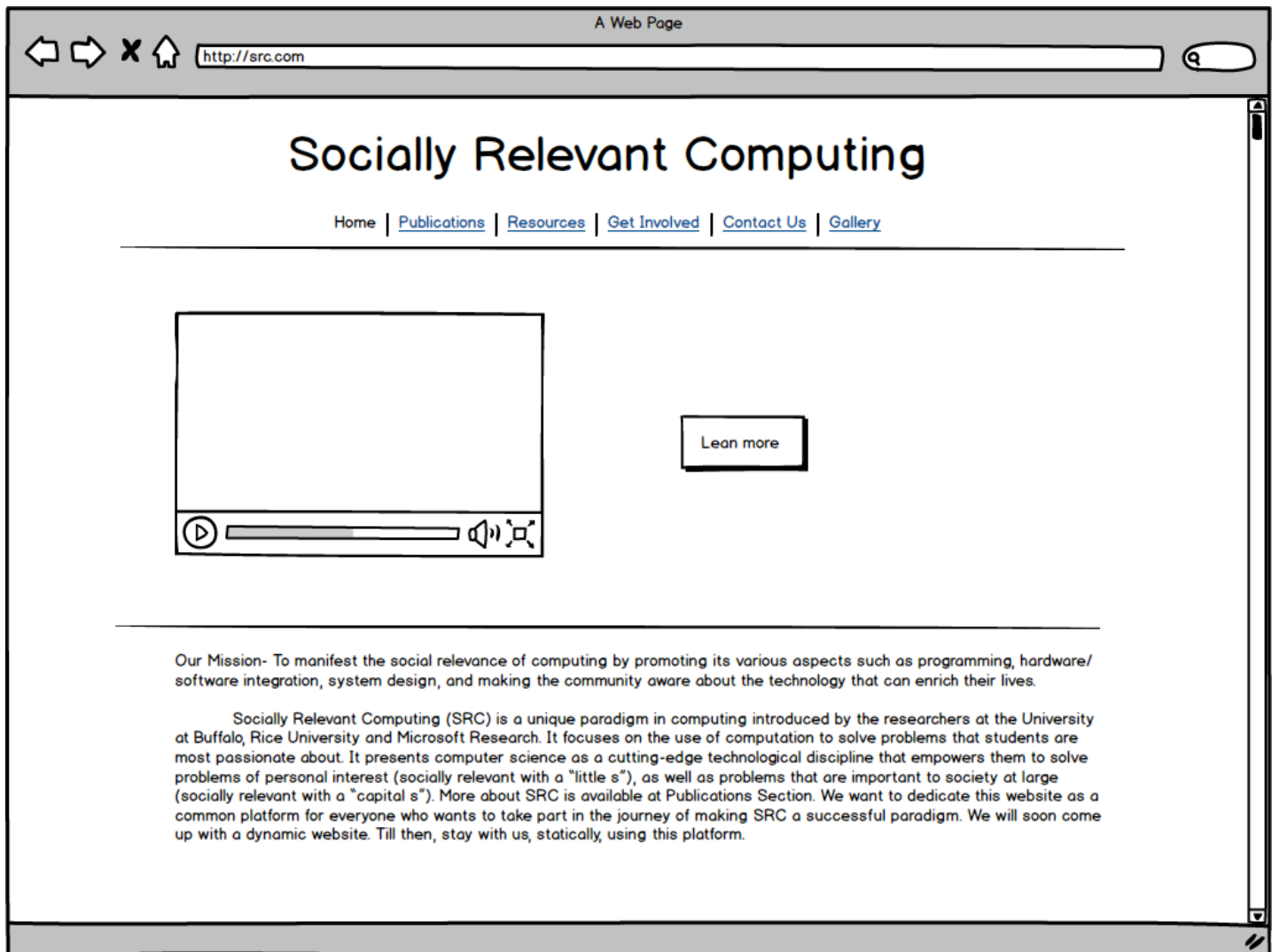
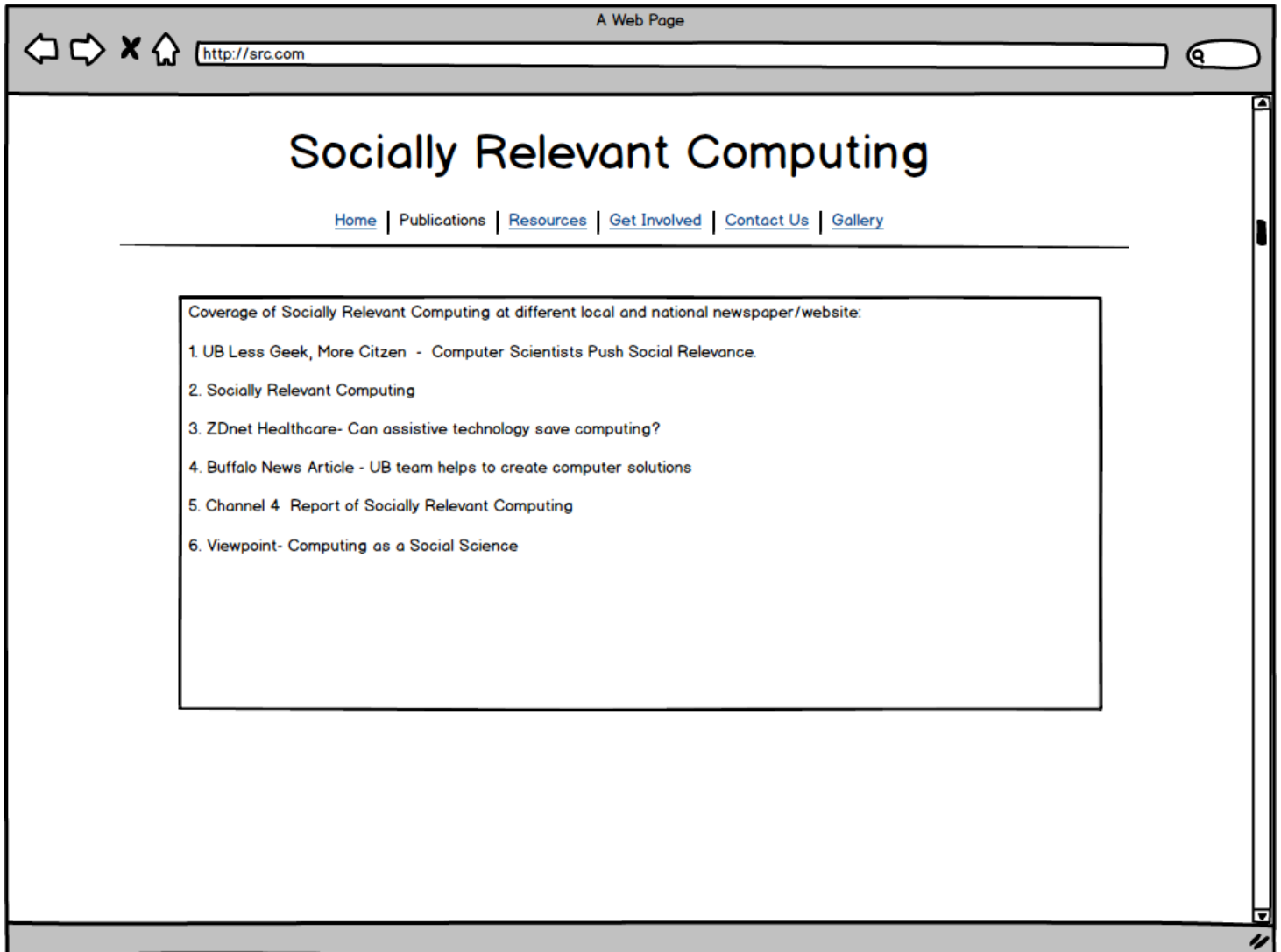
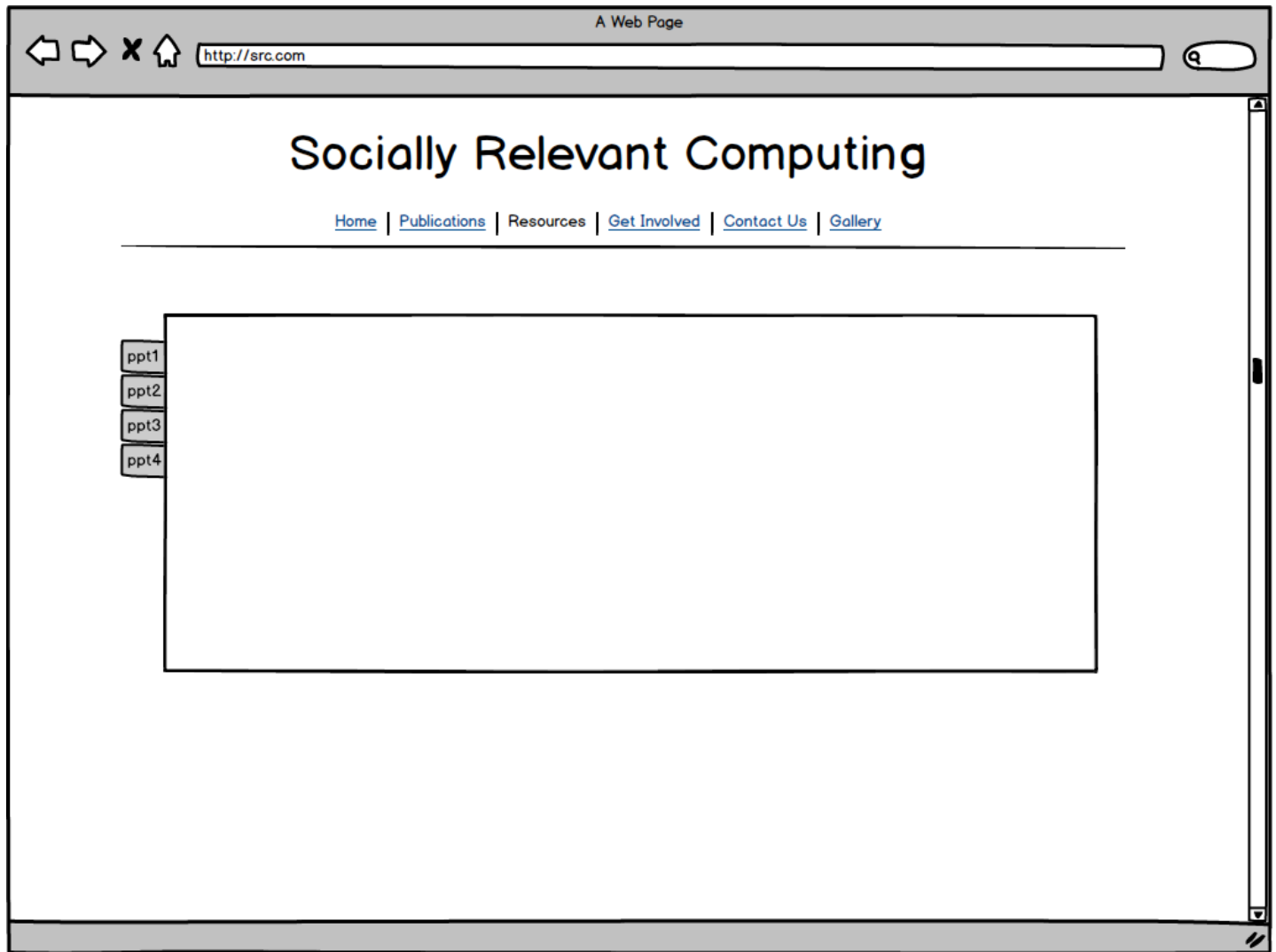


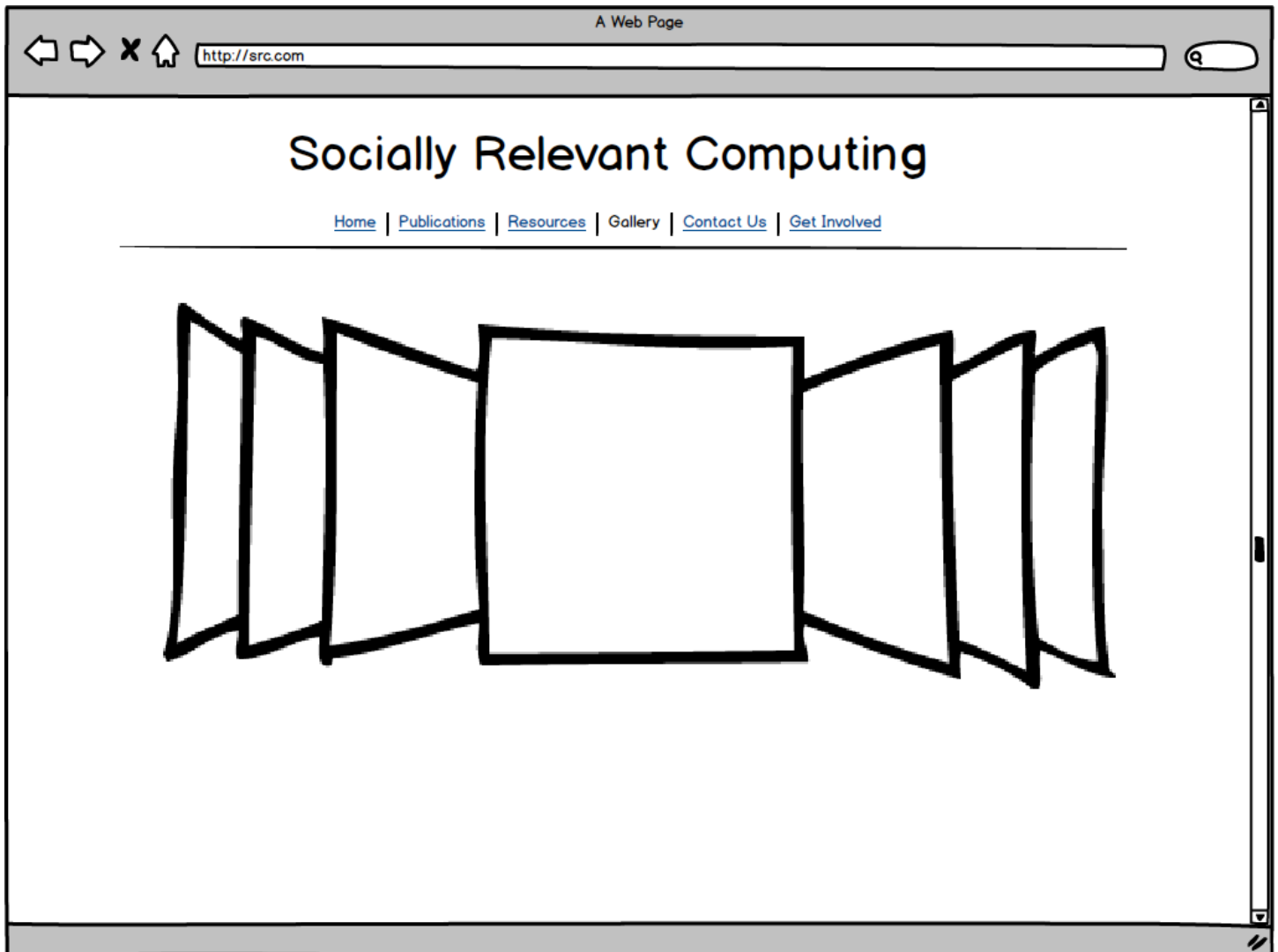
Figure 5 - Payment Gateway

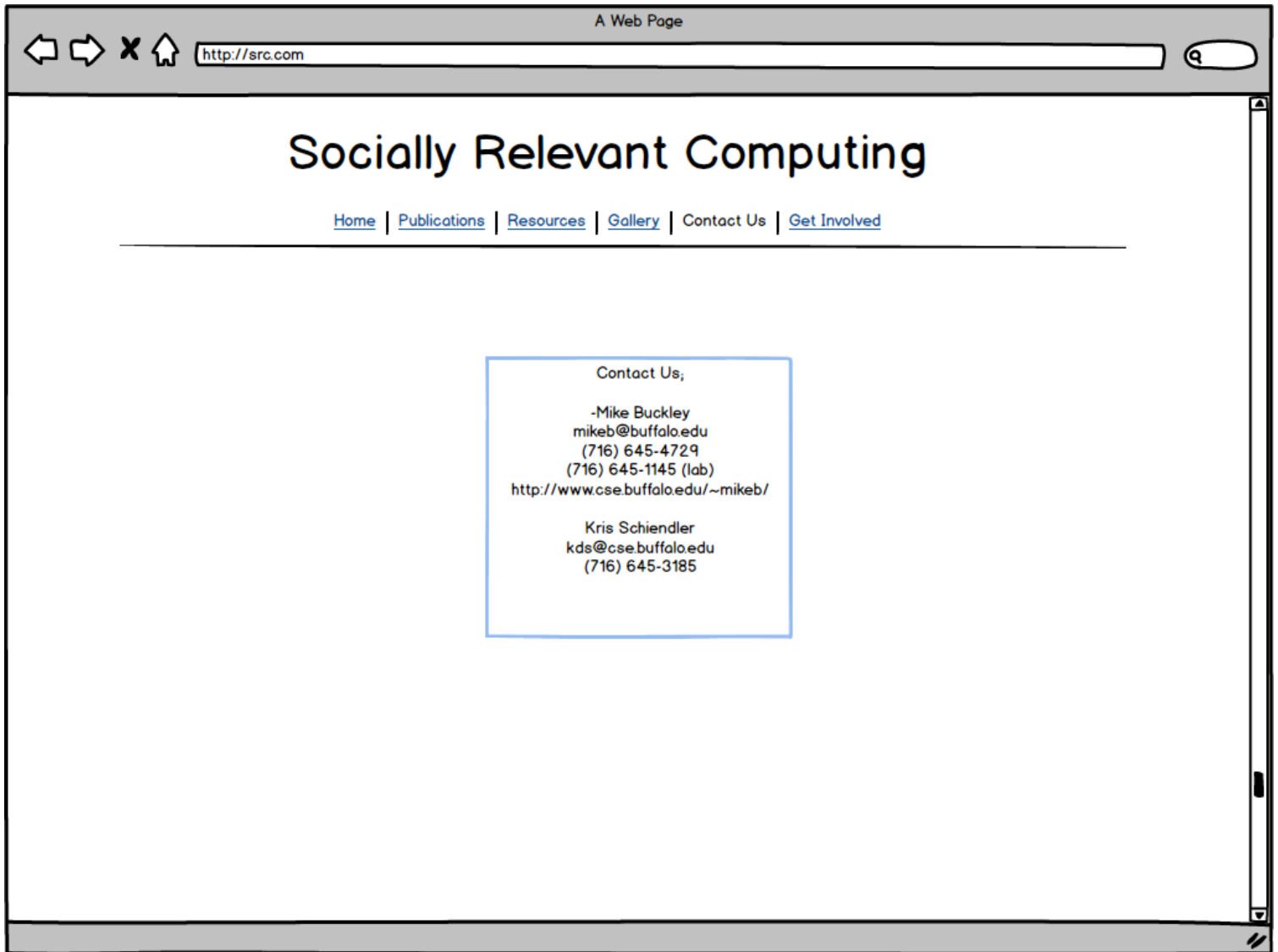
User Screens

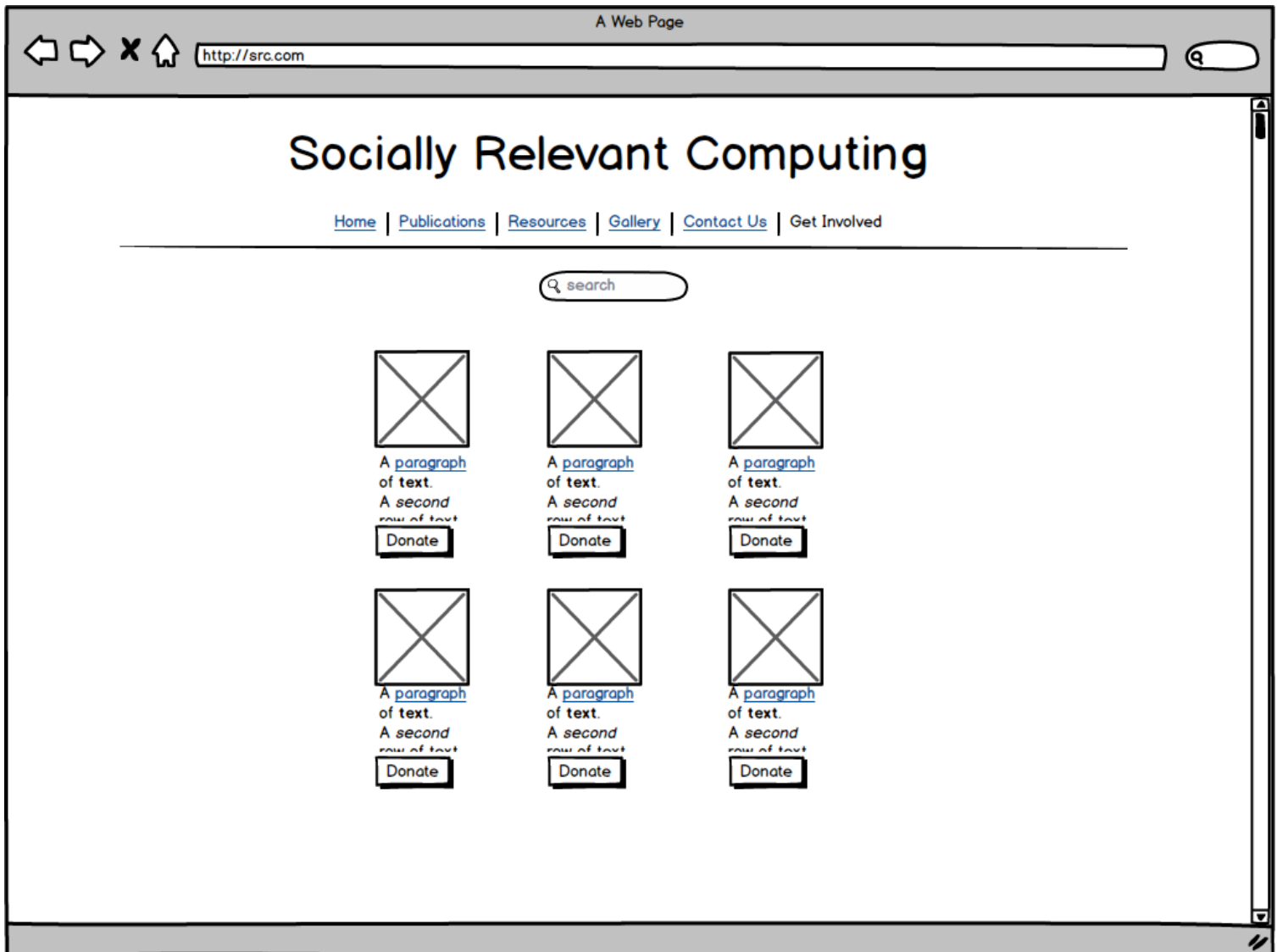










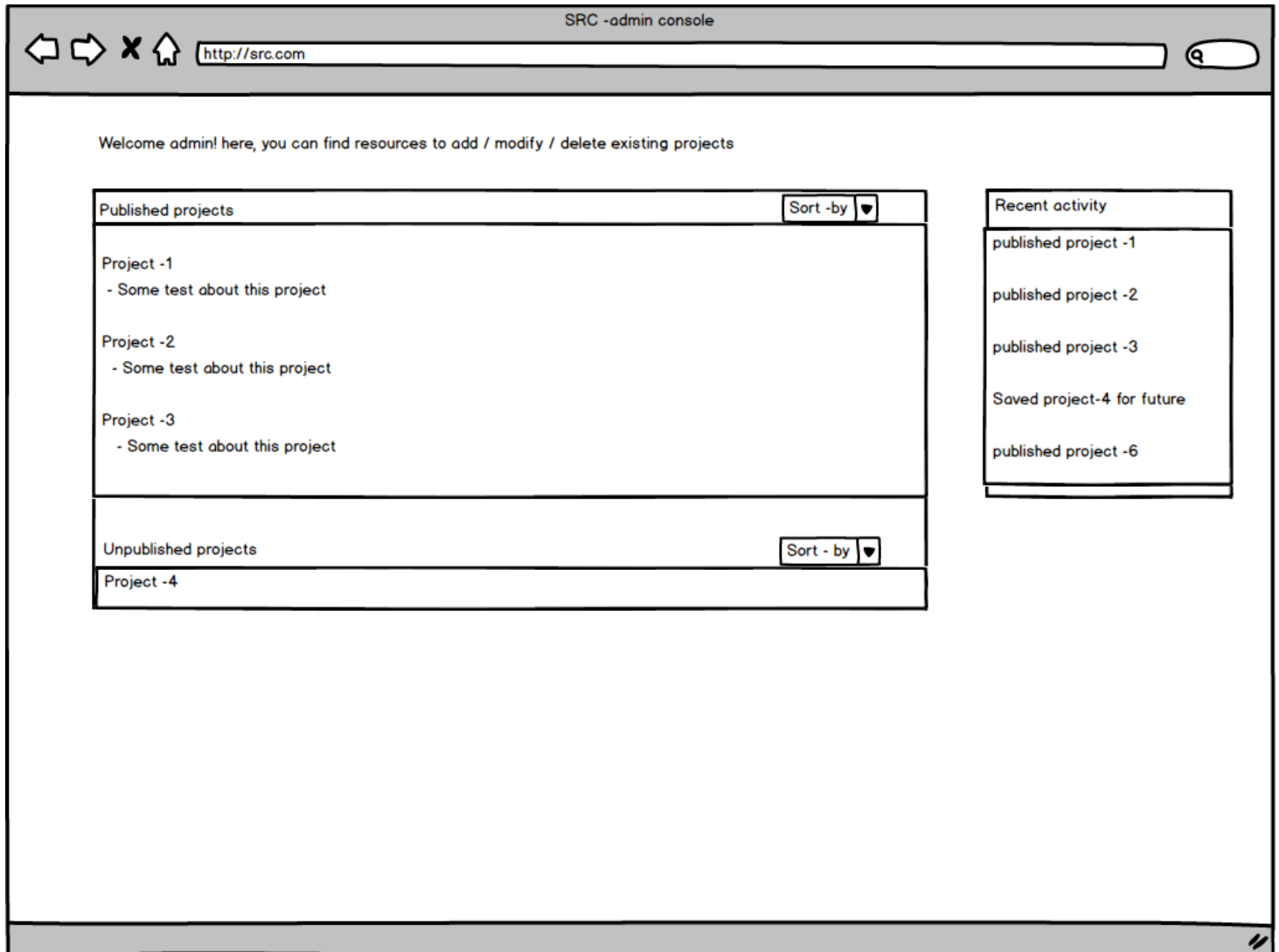


The diagram illustrates a web browser window for an administrative console. The browser's title bar is labeled "SRC - admin console". The address bar contains the URL "http://src.com". The main content area features a centered login form. The form is titled "LOGIN" and includes two input fields: one for "username" and one for "password".





LOGIN

username


password



SRC - admin console



http://src.com

















modify this project

project title :

project -4

project description




This project talks about how humans can make mock ups using this awesome new tool called I Balsamiq

Date

/

/



[Delete this project](#)

Tags

mockups,

Recent activity

published project -1

published project -2

published project -3

Saved project-4 for future

published project -6

Dependencies, Constraints & Limitations

Assumptions

- We assume that the material required for the projects to be included is already provided.
- The server will be initially hosted from university server, later it will be moved to independent server.
- Currently, University at Buffalo and Rice University are the only two participants. All other universities can participate by contacting the administrator.
- The primary goal is to show how computer science can be applied to solving problems in social challenges.
- Anyone who is interested is free to support a project by donating for it and spread the word across social networks.

Limitations / Constraints

- Sponsors will be funding the projects based on their interest, so the projects need not be executed in the time order they posted. There might be an indeterminate delay.
- This concept should be marketed well in order to get more sponsors. In this way there will be a momentum to make it bigger.
- Maintenance of the website is not only a major constraint but increases cost as the database requires some high quality servers as the SRC grows bigger.
- If sponsors do not support a project then there will be difficulties in proceeding further with it. As there is no monetary regulation on the users, there won't be a suitable replacement to sponsors for projects involving lot of money.
- The primary goal of this project is to provide an interactive platform for students, corporates and organizations. Also advertisements might be allowed, as they will increase the revenue.
- Though the website is designed in a user-friendly way, we assume that a basic computer and Internet knowledge is required for users.
- Collecting the materials required for the course and projects available from various universities, online resources would require coordination among people involved.
- All the backer credentials will be stored in database.

Request form for changes to SRC website

If you would like to make any changes to the project description, including new features, feature cuts, limitations, etc., please fill out this form send it to talkingheadcompanion@gmail.com as soon as possible.

Customer Information:

Name:

Email Address:

Phone #:

User Profile:

Organization:

Date:

Type of Change:

☐ Functional ☐ User Interface ☐ Other (specify)

Priority:

☐ High ☐ Medium ☐ Low

Impact:

☐ High ☐ Medium ☐ Low

Reasons for change:

Describe required change:

Cross Reference Listing:

No.	Module	System Specification	Software Requirement Specification
1	Static Pages	Functional requirements (page 6)	Page 6,7,8
2	Admin Interface	Functional requirements (page 6)	Page 6,7,8
3	Projects Page	Functional requirements (page 6)	Page 6,7,8
4	CMS Functionality	Functional requirements (page 6)	Page 6,7,8
5	Functional Requirements	Page 6	Page 6
6	SRC Website management	Page 6	Page 9
7	Funding a project	Page 6	Page 11
8	Mobile phone integration	Page 6	Page 6
9	Notifications	Page 6	Page 6
10	Donation	Functional requirements (page 6)	Page 11