

The Simple Rostering System

Ben Griffiths (4302927)

COMP3760 (6 units)

Supervisor: Eric McCreath

Outline

- Background & Motivation
- What is the Simple Rostering System?
- Demonstration
- Design & Implementation
- Evaluation
- Conclusion



What is a roster?

An example

	Monday	Tuesday	Wednesday	Thursday	Friday
Bins	Tony	Anne	Tony	Anne	Fred
Dishes	Fred	Tony	Fred	Tony	Anne
Lawn	Alex	Alex	Alex	Joel	Joel
Bathroom	Joel	Joel	Fred	Fred	Fred
Kitchen	Anne	Tony	Anne	Anne	Tony

Household Chores

What is a roster?

Another example

	Monday	Tuesday	Wednesday	Thursday	Friday
Cynthia					
Alfred					
John					
Tina					
Alice					

Supervision duty at a school

What is a roster?

Staff Morning Tea

21st May – Carol

28th May – George

4th June – Gary

13th June – Carol

20th June – Gary

27th June – George

3 Basic Elements

- Task(s)
- People
- Dates/Times

What's Wrong With This Picture?

- These rosters are typically printed out and displayed somewhere
- A little “20th century”?
- Sometimes use spreadsheet or word processing software, for layout only
- Existing rostering software is made for businesses – expensive, many complex features

How Can We Make It Better?

- Web services



- Ubiquitous email



- Automation

- Waste reduction



- Overall - greater efficiency and effectiveness

The Simple Rostering System

- Web-based rosters
 - Creating
 - Storing
 - Sharing
- Invite participants by email
- Email reminders and notification of changes
- Uses LAMP (Linux, Apache, MySQL, PHP)

Scope/Objectives

- Demonstrate that a system can feasibly be developed to fill this niche
- Lay the groundwork for future development
- Make simple rosters easier & more efficient
- Developing the user interface was beyond the scope of the project

User Story

- 2 employees in an office environment
- Sharing responsibility for staff kitchen
- Want to allocate regular tasks:
 - Wipe down benches
 - Wash dishes (in dishwasher)
 - Clean microwave
- Email reminders, and swapping/changing tasks between employees is desirable

Demonstration

View Roster - The Simple Rostering System - Windows Internet Explorer

http://www.nebwebaustralia.com/srs/index.php?area=roster&task=view&id=1

File Edit View Favorites Tools Help

NW View Roster - The Simple Rostering System

THE SIMPLE ROSTERING SYSTEM

View Roster

Home Rosters Logout About

Kitchen Duties

Start: 09/06/2008
End: 13/06/2008

Tasks

- 09/06/2008: Wipe down benches (Alice)
- 10/06/2008: Wipe down benches (Wally)
- 11/06/2008: Wipe down benches (Alice)
- 12/06/2008: Wipe down benches (Wally)
- 13/06/2008: Wipe down benches (Alice)

Send reminder Day(s) before all my tasks in this series of rosters.

[Back to roster list](#)

Developed by NebWeb Logged in as Alice

Done Internet 100%

Design & Implementation

- LAMP architecture
 - Inexpensive to set up (or acquire access to)
 - Widespread (easier for people to customise)
- Web-based interface
 - Accessible to a wide variety of users
- Simplicity
 - Even with few features, the system still encompasses a wide variety of uses
 - Less features makes the system easier to use

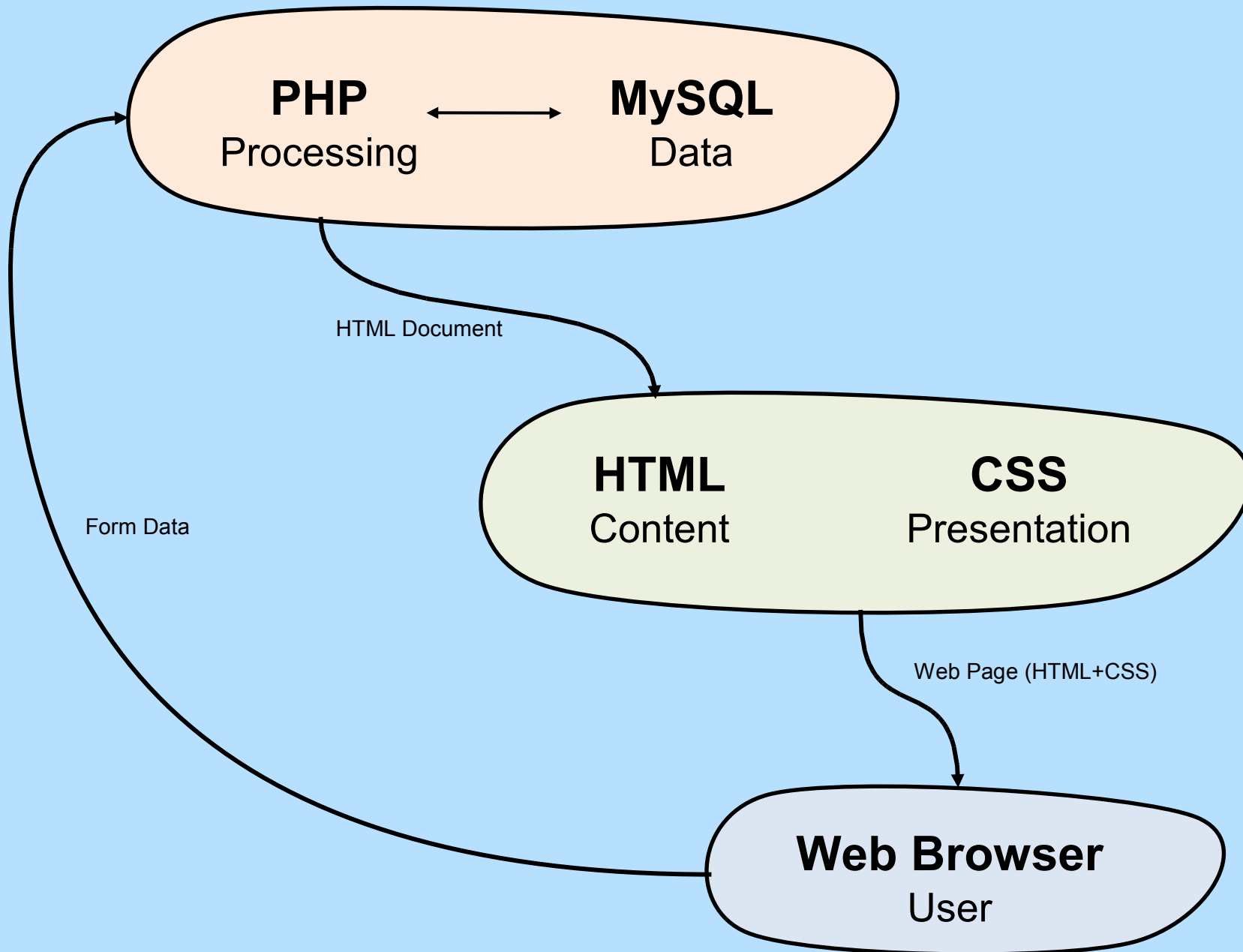
Design Separation of Concerns

- System framework
 - Database connection
 - Page construction
 - Global preferences
 - Emailing
- User Accounts
 - Invitation, registration, deletion
 - Authentication, session management
- Rosters
 - Creation, modification, deletion
 - User participation
 - Reminders/notifications

Data Representation

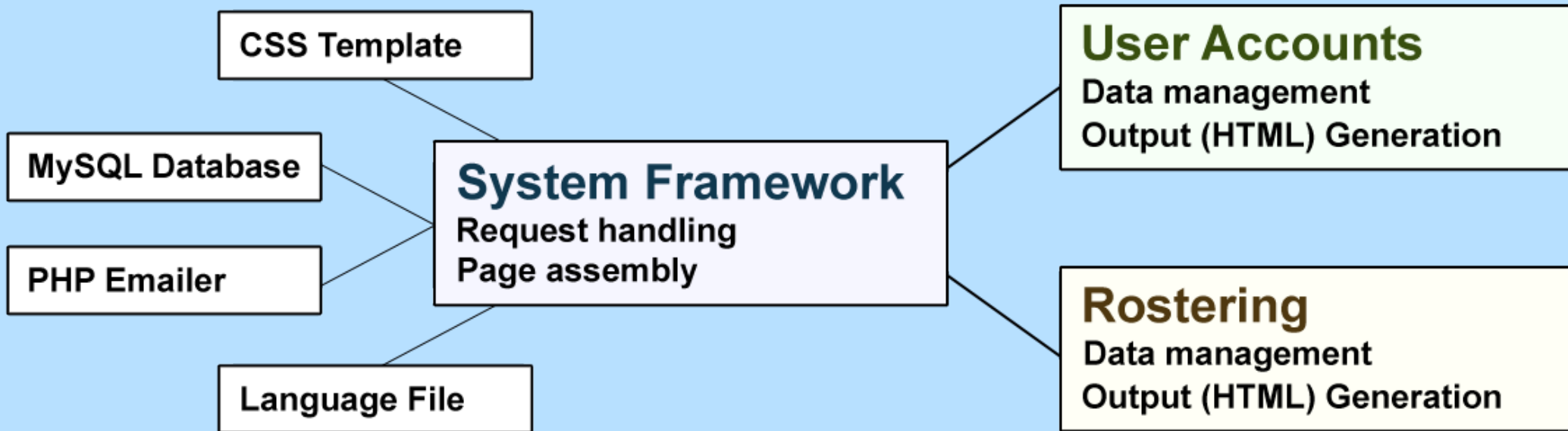
- Series as a sequence of Rosters
 - Rosters can repeat at Intervals within a series
- Roster as a collection of Tasks
- Tasks as an archetype for Task Instances
 - Task Instances can repeat at Intervals for a particular Task
- Task Instances as a date when a task is to be performed
- User participate in Rosters
- Users are assigned Task Instances

Implementation Technologies

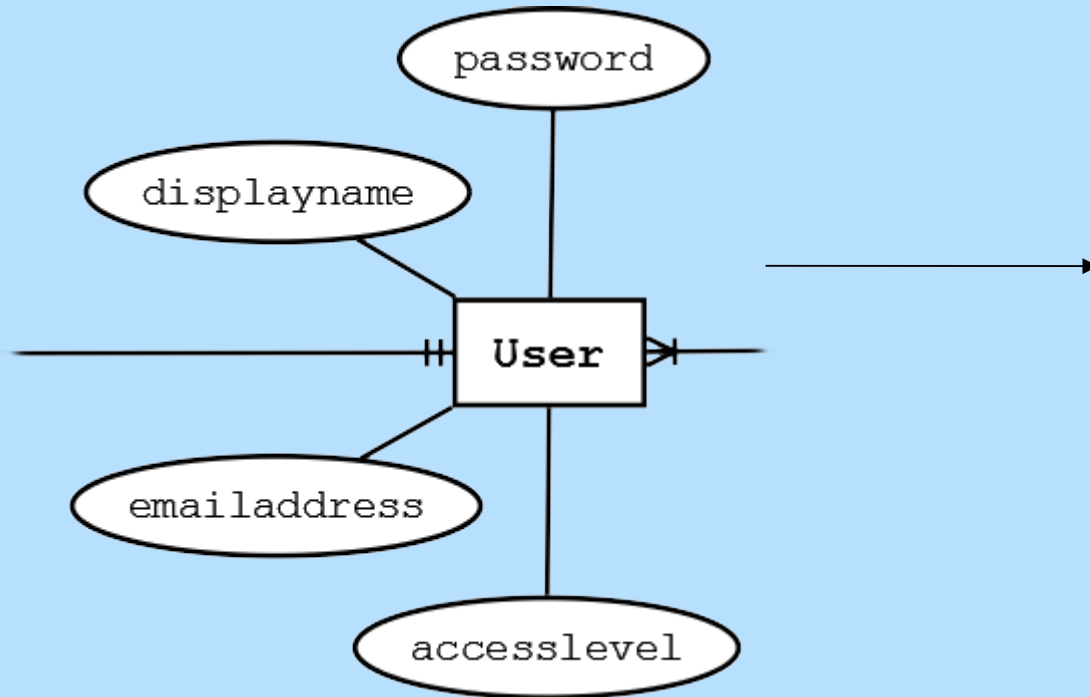


Implementation

PHP Code Structure



Database Tables



srs_user

id : Integer

emailaddress : VarChar

access : TinyInt

password : VarChar

active : TinyInt

displayname : VarChar

Evaluation

- Integration testing:
 - Perform tasks through the user interface
 - Verify correct result through user interface
 - Verify correct result in database
- Comparison to existing systems:
 - Evaluate system performance for specific scenarios
 - Compare to paper-based roster, and commercial rostering software
- Focus on functionality rather than interface

Evaluation Testing Results

Test Suite - 01/06/08

Task to attempt	Bugs found (FE/DB)	Bugs fixed
Invite a new user	1	1
New user registration	1	1
Log in (correct details)	0	0
Attempt log in (incorrect details or code injection)	0	0
Log out	0	0
Reset password	0	0
Create new roster	1	0
Invite user to roster	3	3
Remove user from roster	1	1
Edit and save roster	1	1
Create a new task for roster	1	1
Edit and save task for roster	2	2
Remove task from roster	2	2
Assign user to task instance	1	1
Re-assign task instance to another user	0	0

Challenges

- Data representation – flexibility with manageable complexity
- Interface design – the same problem (not addressed in this project)
- Predicting scope and managing workload



Conclusion

- It is feasible to build a simple rostering system
- Rosters do not fit easily into typical data representations – numerous co-dependent times and dates
- A simple system must have a simple user interface
- This project provides the groundwork for developing the system to it's potential

Future Work

- Development of an accessible and easy to use web interface (eg. Interactive calendar)
- Potential open source release for community development
- Enhanced availability specification and automated task allocation
- Adaptation into a plugin/component for content management systems (eg. Joomla, Drupal)

The End

Questions?