1. Scalability: (Jan.)
2. Agile Practices
3. Ecology/Mashups
4. Browser/Client
5. Data/Server: (Feb.)
6. Security/Privacy
7. Analytics*
8. Cloud/Map-Reduce
9. Publish APIs: (Mar.) *
10. Future
   * assignment due
Administrative Stuff

• Advocates, send a team roster to staff
  • keep us up to date with any issues
• MySQL DB accts set up for each project
  • will send username/password
• Be checking in code already
  • extra credit: runnable project now
Project Status Update

• Cloned 4 of 7 project repos
• Saw 2 had significant activity already
• Project members should have 1 checkin
• Extra credit
  • contribute ‘how to get started’ info
Sample Project History

gigs merge
Merge branch 'master' of ssh://imyth.stanford.edu/afs/sir.stanford.edu/class/cs193s/git/student/services
now http://localhost:8888/Works
reconfigured to stanford db
adding some controller logic
adding gig model changed
Adding some controller logic
gig_model
Merge branch 'master' of ssh://imyth.stanford.edu/afs/sir.stanford.edu/class/cs193s/git/student/services
Added stuff to gig model, static constants to use throughout
database setup!
database...it works!
deleting message model
Merge branch 'master' of ssh://imyth.stanford.edu/afs/sir.stanford.edu/class/cs193s/git/student/services
Updated database.php config.
A new sql file was created to allow phpmyadmin to run the query.
Set up CI to connect to our DB on every page load.
Updated database.php config to connect to our DB online.
Merge branch 'master' of ssh://imyth.stanford.edu/afs/sir.stanford.edu/class/cs193s/git/student/services
Added gig and message models
Adding User model
Testing controller / view
testing gitX
changing base_url for local testing
adding mockups (balsamlq with screenshots)
Merge branch 'master' of ssh://imyth.stanford.edu/afs/sir.stanford.edu/class/cs193s/git/student/services
Checking in basic homepage template to work with, including some js and css files. Also adding a logo image to stimulate discussion.
Added the SQL commit script with the _modified_by_ cliff suffix. Removing the DB name from each of the queries fixed our issue. The DB schema is now in place on the server.
Updating the database schema diagram with notes.
Merge ssh://imyth.stanford.edu/afs/sir.stanford.edu/class/cs193s/git/student/services
Added my name, Ahmed Bashir, to the README
Adding database schema diagram and SQL create script
Adding my name to the README
adding code igniter to repo
Added my name - Clifton
jack rules
added
Adding name to team
Initializing the repository
Privacy in Web Apps

- Prevent identity leaks
  - time out cookies, enforce boundaries
- Limit spam & phishing opportunities
  - no broadcast friending & messaging
- Set policy for data export
  - verify partner policies are compatible
Identity Leaks

- Keep browser code clean
- single user info in cookies
- critical data filtering is serverside
- Prevent other browser window access
- flash can read entire browser DOM
- debuggers are equally powerful
**Membership Abuse**

- Set clear user conduct standards
- develop abuser profile
- Limit access to group broadcasting
- karma points
- Design to promote desired behavior
- react quickly to user complaints
Third Party Data Abuse

- Partner companies’ privacy policies
- similar or more restrictive
- Acquisition, liquidation data policy
- new owners often eager to sell
Value of Social Data

- Phishing is the most common attack
- Social network data allows targeting
- Targeting can be extra precise
- Dramatic increase in attack outcomes
- Online reputation measurement coming
- Web identity theft can be crippling
Plaxo Example

- Original idea
  - global contacts directory
- Implementation
  - trick users into opening address book
  - subvert addressees too
- Most people in system without consent
Security in Web Apps

- Prevent losses across API boundaries
  - fewer data flow paths are better
- Limit resource abuse
  - avoid single user denial of service
- Regular analysis of user activity logs
  - learn to identify unusual patterns
API Boundary Attacks

- Bad data inputs
  - wrong types, text escapes
  - SQL, javascript injection
- Bad transmission & outputs
  - minify / encrypt communications
  - poor error messaging
Resource Abuse

- Resource intensive actions
  - high speed data/messaging requests
  - repeated login/logout cycles
- Boundary overflows
  - large memory uses
  - parameter limits
User Patterns

- Attacks rarely succeed immediately
- Track accounts, IP addresses, cookies
  - correlate where possible
- Log code exceptions, data issues
  - associate with above IDs
Reverse Engineers

- Test every available input in apps
  - character range, length
- Check result page source
  - debuggers make testing quicker
- Try to trigger DB errors, server failures
  - diagnostic pages must never display
Examples

- **Cross Site Scripting**
  - INPUT: `<SCRIPT SRC=http://ha.ckers.org/xss.js></SCRIPT>`

- **SQL injection**
  - query = "SELECT * FROM users WHERE username = " + name + ";"
  - INPUT: "x';DROP TABLE users; SELECT ""
Exception Handling

- General rule for user interaction errors
  - maintain browser state, or
  - show a minimal error message, then
  - allow user to retry (a few times),
  - finally, fail gracefully
- Avoid debug or diagnostic data release
Exception Handling (2)

- Exceptions are causes of resource leaks
  - release DB connections, file handles
- Short startup times make a difference
  - store configurations in a file or DB
- hardcoding complicates updates
Worth Checking Out

- 25 Dangerous Programming Errors
  - http://cwe.mitre.org/top25/

- Cross Site Scripting Cheat Sheet
  - http://ha.ckers.org/xss.html
Q & A Topics

- Anonymity vs. Privacy
- User age and legality of contracts
  - age of license, 16 years, 13 years