Finish Communications
Logging In
Systems
Server Sent Events

- Long Polling without setup/teardown of connection
- Same server-side concerns for open connections
- ONE WAY!!
  - Only time client says anything to server is at setup
  - GET only
Server-sent events

- javascript EventSource
- need to write “handlers” for message types
- default handlers
- PHP
  - message syntax is important but otherwise can look kind of like a long poller, except:
    - Syntax of response is special
    - After sending one response, the return to the loop and send another ...
  - “retry” is a poll-like thing

```php
// File: eventer.html

echo "event:basic\n";
echo "data:" . json_encode($a) . "\n\n";
echo "retry:" . ($reload*1000) . "\n";
ob_flush();
flush();
```

File: eventer.html
evented.php
Fully symmetric communication

Web Sockets

- Node is most widespread implementation
  - Usually called Node.js
- On server side, NOT apache, or Nginx, ..
- low-latency, full-duplex communication makes the location of code less important

For much more: https://www.youtube.com/watch?v=jo_B4LTHi3I
• JavaScript Object Notation
  • Nothing really to do with javascript
  • A way of encoding arrays and dictionaries
  • PHP and Javascript both have encode and decode functions
  • Certainly the best way to communicate when polling/server-send events from PHP to Javascript
  • Can also send from Javascript to PHP

```php
<?php
$a = [1,2,3,"aa", "bb", "aa"];
echo json_encode($a);
echo "\n";
$b = ["a" => "b", 2 => 3, 4 => "q", "b" => [1,2,3,4]];
echo json_encode($b);
echo "\n";
?>
/** Normally only return a single JSON object from PHP **/

[1,2,3,"aa","bb","aa"]
{"a":"b","2":3,"4":"q","b":[1,2,3,4]}
Sending and receiving JSON

```javascript
const jsonn = {
  "email": "hey@mail.com",
  "password": "101010"
};

const dataToSend = JSON.stringify(jsonn);

function sendFetch() {
  fetch("sendJSON.php", {
    method: "post",
    headers: { "Content-Type": "application/json" },
    body: dataToSend
  }).then(function(response) {
    response.text().then(function(txt) {
      console.log(txt);
      json = JSON.parse(txt);
      console.log(json["email"]);
    });
  });
}

function sendJQ() {
  let posted = $.post("sendJSON.php", dataToSend);
  posted.done( function(data) {
    console.log(data);
    console.log(data["email"]);
  });
}
```

```php
<?php
// Handling data in JSON format on the server-side using PHP
header("Content-Type: application/json");
// build a PHP variable from JSON sent using POST method
$v = json_decode(stripslashes(file_get_contents("php://input")));
$v->hello='geoff';
$vv = (array)$v;
$vv["success"] = true;
echo json_encode($vv);
?>
```
Logging In

- Lots of systems require logins
- There are infinite approaches to this. I present one
- Requires at least
  - 1 html page
  - 1 php page
  - 1 database table

```sql
create database if not exists login;
use login;
drop table if exists users;
create table users (
    userNumber int not null auto_increment,
    loginname varchar(64) NOT NULL,
    passwd varchar(94),
    sectionid int,
    primary key (userNumber)
);
```

Allow one table to support multiple pages
Adding Users

• Requires a whole extra system

• In this case, I decided not to bother with html ...

• So just a php program

```php
<?php
// Use from command line php -f adduser.php name=XXXX pass=YYYY
// where both XXXX and YYYY are clear text

foreach ($argv as $arg) {
    $e=explode("=",$arg);
    if(count($e)==2)
        $_GET[$e[0]]=$e[1];
    else
        $_GET[$e[0]]=0;
}

//Standard DB connection stuff here

$sql = "INSERT into users (loginname, passwd, sectionid) VALUES ('" .
$_GET["name"] . ", '', ".password_hash($_GET["pass"],PASSWORD_BCRYPT) . ", 1);"

echo $sql;
echo "\n";
$result = $conn->query($sql);
?>
```
Login page

• Users hate page reloads so I use a lot of CSS and Javascript

• There are lots of opinions about “remember me” and how to implement
  • I chose a server side solution that does not save password
Server Side of login

getSavedLogin.php

```php
<?php
session_start();
if (array_key_exists("username", $_SESSION)) {
    echo $_SESSION["username"];
} else {
    echo "unknown";
}
?>
```

loginpage.php

```php
function login_check() {
    $sql = "select passwd from users where sectionid=1 and loginname='" . $_REQUEST["uname"] . "';";
    $result = $conn->query($sql);
    if ($result) {
        $row = $result->fetch_assoc();
        if (password_verify($_REQUEST["passwd"], $row["passwd"])) {
            $_SESSION["loggedin"] = true;
            $_SESSION["sectionlog"] = 1;
            if ($_REQUEST["remember"]) {
                $_SESSION["username"] = $_REQUEST["uname"];
            } else {
                unset($_SESSION["username"]);
            }
        } else {
            return "success";
        }
    } else {
        $_SESSION["loggedin"] = false;
        unset($_SESSION["sectionlog"]);
        return "bad password";
    }
} else {
    $_SESSION["loggedin"] = false;
    unset($_SESSION["sectionlog"]);
    return "bad username";
}
```
<?php

function start_update_session() {
    if (session_status() != PHP_SESSION_ACTIVE) {
        session_start();
    }
    $now = time();
    if (isset($_SESSION['discard_after']) && $now > $_SESSION['discard_after']) {
        // this session has worn out its welcome; kill it and start a brand new one
        session_unset();
        session_destroy();
        session_start();
    }
    $_SESSION['discard_after'] = $now + 3600;
}

function good_session() {
    if (array_key_exists("loggedin", $_SESSION) && $_SESSION["loggedin"] && in_array("sectionlog", array_keys($_SESSION)) && in_array("sectionlog", array_keys($_SESSION)) == 1) {
        return true;
    } else {
        return false;
    }
}

?>

<html>
<body>
<?php
    start_update_session();
    $result = login_check();
    if ($result == "success") {
        header("Location: nextpage.php", true, 301);
        //include("nextpage.php");
    } else {
        header("Location: modallogin.html", true, 301);
    }
?>
</body>
</html>
Systems
Towers of Hanoi

• Goal: provide a way for 206 students to have friendly competition with the towers of Hanoi
• Design: a stopwatch with Hanoi instructions
• A backend to generate a graph of times.
• Graph drawn on html canvas object

Files: towers.html, aaa.php

```sql
create database if not exists hanoi;
use hanoi;
drop table if exists timedata;
create table timedata (
    id int NOT NULL auto_increment primary key,
    actor varchar(64),
    witness varchar(5),
    time varchar(10)
);
```