# COMP1011 – Advanced Web Programming

# Project 2

## Final Team Project

Due class #14 (April 22nd, 2013) @ midnight

Value 35%

Final Team Project **Maximum Mark: 55**

**Overview**: Working with a partner (or on your own) and utilizing your accumulated knowledge of web technologies including HTML5, CSS3, PHP, MySQL, JavaScript and jQuery create a web app from one of the following App templates:

- Survey Site
- Simple Social Network Site
- Web Forum
- Bracket / Tournament Web App
- Incident Management Web App
- Blogging Site

Your Web App **must be hosted on a live site** (the school server is acceptable). You will have core functionality requirements as well as specific criteria for your type of Web App.

## Instructions :

***COMMON REQUIREMENTS (25 MARKS)***

Each Web App will share these common requirements:

**(5 Marks: Site Structure, 5 Marks: Internal Documentation, 10 Marks: External Documentation, 5 Marks: Version Control)**

1. Your Web App must be designed and structured using well semantic HTML, CSS, JavaScript and jQuery (front end technologies) along with a dynamic back-end (using PHP and MySQL) **(5 Marks: Site Structure):**
   a. Your Site must either be **responsive,** and adapt to various viewports or have an additional **mobile specific** user interface designed with jQuery Mobile or other mobile frameworks (1 Marks: Site Structure).
   b. Your **CSS** rules reside in separate file(s) in their own folder and adhere to best practices (1 Mark: Site Structure).
   c. Your **JavaScript** files (and other external code) are contained in their own folder and are appropriately linked to your site. (1 Mark: Site Structure).

    d. Your **images and multimedia assets** are contained in their own folder and appropriately linked to your site. Your multimedia assets may need to be modified for mobile (1 Mark: Site Structure).
    e. All Your Code (HTML, CSS, JavaScript, jQuery, jQuery Mobile etc.) is error free (1 Mark: Site Structure).

2. Include **Internal Documentation** for your site **(5 Marks: Internal Documentation):**
    a. Ensure you include a **comment header** for your **html**, **CSS, JavaScript and PHP files** that indicate: the **File name**, **Author's name**, **web site name**, **file description** (2 Marks: Internal Documentation).
    b. Ensure you include a **section headers** for all of your **html structure, CSS style sections,** and any **functions** (1 Marks: Internal Documentation)
    c. Ensure all your code uses **contextual variable names** that help make the files human-readable (1 Marks: Internal Documentation).
    d. Ensure you include **inline comments** that describe your GUI Design and Functionality. (1 Marks: Internal Documentation)
3. Create an **External Document** for your Web App that includes **(10 Marks: External Documentation)**:
    a. A **company Logo** (1 Marks: External Documentation).
    b. **Table of contents** (1 Marks: External Documentation).
    c. A **Detailed description** of your Web App including its core functionality (3 Marks: External Documentation).
    d. A **Wireframes Section** that include a wireframe image and appropriate arrows and labels for each page template of your Web App (2 Marks: External Documentation)
    e. **Screen Capture Section** that includes Screen Shots (samples) of each of your site's templates. (2 Marks: External Documentation).
    f. **Potential Future Functionality –** a section that describes features that could be added to your app but do not as yet exist given the time constraints. (1 Mark: External Documentation).
4. Share your files on **Github** for this assignment to demonstrate Version Control Best Practices **(5 Marks: Version Control).**
    a. Your repository must include **your code** and be well structured (2 Marks: Version Control).
    b. Your repository must include **commits** from both you and your partner that demonstrates the project being updated at different stages of development – each time a major change is implemented (3 Marks: Version Control).

## *APP SPECIFIC REQUIREMENTS (30 MARKS)*

SURVEY SITE (OPTION)

**(15 Marks: GUI, 15 Marks: Functionality)**

1. User Management and site security **(5 Marks: GUI, 3 Marks: Functionality):**
    a. **User Registration** must be included. A form will allow the user to enter profile information (**username**, **password, email address, etc)** which will be stored in a database table (2 Marks: GUI, 2 Marks: Functionality).
    b. The user will be able to **Login**, **Logout** and **modify** his or her profile (1 Mark: GUI, 1 Mark: Functionality).
    c. **Site security** will prevent non-registered users from **creating** a survey or entering secure areas of the site (2 Marks: Functionality).
2. Users can **Create** a Survey **(6 Marks: GUI, 6 Marks: Functionality):**
    a. After a user is **registered** and **logged in**, he or she can create a survey based on 1 or 2 possible **survey templates** (e.g. Multiple Choice, Agree/Disagree, Short Answer, etc.) (3 Marks: GUI, 3 Marks: Functionality).
    b. The user should be able **customize survey questions**. This includes the question text and response options (2 Marks: GUI, 2 Marks: Functionality).
    c. The user should be able to create a **lifetime** for the survey (i.e. when the survey becomes **active** and when it **expires**) (1 Mark: GUI, 1 Mark: Functionality).
3. Anonymous users can **Respond** to any active survey **(2 Marks: GUI, 3 Marks: Functionality):**
    a. Anonymous users should be able to **select** an active survey and respond to survey questions. (2 Marks: GUI, 2 Marks Functionality).
    b. Survey responses will be stored in the database for later use (1 Mark: Functionality).
4. Secure Reporting Section **(2 Marks: GUI, 3 Marks: Functionality):**
    a. A **registered user** will be able to get simple analysis for any survey that he or she **owns** including **number of respondents** and survey answer **statistics** (1 Mark: GUI, 2 Marks: Functionality).
    b. The statistics from each survey can be **exported** in some manner (e.g. **emailed**, **printed**, exported to **excel**, etc.) (1 Mark: GUI, 1 Mark: Functionality).

SOCIAL NETWORK SITE (OPTION)

**(15 Marks: GUI, 15 Marks: Functionality)**

1. User Management and site security **(5 Marks: GUI, 3 Marks: Functionality):**
    a. **User Registration** must be included. A form will allow the user to enter profile information (**username**, **password, email address, etc)** which will be stored in a database table (2 Marks: GUI, 2 Marks: Functionality).
    b. The user will be able to **Login**, **Logout** and **modify** his or her profile (1 Mark: GUI, 1 Mark: Functionality).
    c. **Site security** will prevent non-registered users from **connecting with other users,** or entering secure areas of the site (2 Marks: Functionality).
2. Connect / Share with other users **(6 Marks: GUI, 6 Marks: Functionality):**
    a. After a user is **registered** and **logged in**, he or she can share part or all of their profile information with another user. (3 Marks: GUI, 3 Marks: Functionality).
    b. Users that are connected can **view each other's social feeds** (2 Marks: GUI, 2 Marks: Functionality).
    c. A user can choose to **break a connection** with another user (1 Mark: GUI, 1 Mark: Functionality).
3. User Search Functionality **(2 Marks: GUI, 3 Marks: Functionality):**
    a. Registered users should be able to **search for other users**, **select them from a list** and send them an **invitation to connect**. (2 Marks: GUI, 3 Marks Functionality).
4. Post on a user's Social Feed or send email to a specific user **(2 Marks: GUI, 3 Marks: Functionality):**
    a. A **registered user** will be able to **post a comment** on their social feed or another user's feed that they are connected with. The comment will only be viewable by connected users. (1 Mark: GUI, 2 Marks: Functionality).
    b. A **registered user** can send a private message to a specific user (1 Mark: GUI, 1 Mark: Functionality).

WEB FORUM SITE (OPTION)

**(15 Marks: GUI, 15 Marks: Functionality)**

1. User Management and site security **(5 Marks: GUI, 3 Marks: Functionality):**
    a. **User Registration** must be included. A form will allow the user to enter profile information (**username**, **password, email address, etc)** which will be stored in a database table (2 Marks: GUI, 2 Marks: Functionality).
    b. The user will be able to **Login**, **Logout** and **modify** his or her profile (1 Mark: GUI, 1 Mark: Functionality).
    c. **Site security** will prevent non-registered users from posting comments in a forum or creating a new topic (2 Marks: Functionality).
2. Create Forum Topic **(6 Marks: GUI, 6 Marks: Functionality):**
    a. After a user is **registered** and **logged in**, he or she can create a **new Forum Topic.** Each forum topic will present other registered users an area where they can post comments. Any number of Forum Topics can be created. (3 Marks: GUI, 3 Marks: Functionality).
    b. Initially Forum Topics will be presented to users in a clickable list format. (2 Marks: GUI, 2 Marks: Functionality).
    c. When a user creates a forum he is considered the **owner** and can choose to **close** it to any further comments. (1 Mark: GUI, 1 Mark: Functionality).
3. Users Can Post to any Forum Topics **(4 Marks: GUI, 6 Marks: Functionality):**
    a. A Forum Topic will display a **post button** to registered users that allow them to Post a comment (1 Mark: GUI, 2 Marks: Functionality).
    b. When the **post button** is selected, a registered user will **fill out a form** and will be able to add an un-moderated comment to a Forum Topic. The form will have several **required fields** to ensure validation and security (e.g. CAPTCHA) (2 Marks: GUI, 3 Marks: Functionality)
    c. Forum Topics will be visible as a long narrative with the latest comment on the top or bottom depending on the designer’s preference. (1 Mark: GUI, 1 Mark: Functionality).

TOURNEY BRACKETS SITE (OPTION)

**(15 Marks: GUI, 15 Marks: Functionality)**

1. User Management and site security **(5 Marks: GUI, 3 Marks: Functionality):**
    a. **User Registration** must be included. A form will allow the user to enter profile information (**username**, **password, email address, etc)** which will be stored in a database table (2 Marks: GUI, 2 Marks: Functionality).
    b. The user will be able to **Login**, **Logout** and **modify** his or her profile (1 Mark: GUI, 1 Mark: Functionality).
    c. **Site security** will prevent non-registered users from posting comments in a forum or creating a new topic (2 Marks: Functionality).
2. Create A New Tournament **(4 Marks: GUI, 3 Marks: Functionality):**
    a. After a user is **registered** and **logged in**, he or she can create a **new Single Elimination Tournament.** Each Tournament will have a name and include a short description (2 Marks: GUI, 1 Marks: Functionality).
    b. When a user creates a Tournament he is considered the **owner** and can choose when it becomes **active** and when it is **completed**. (2 Marks: GUI, 2 Marks: Functionality).
3. Users Can Register Players **(2 Marks: GUI, 5 Marks: Functionality):**
    a. Users can register players using a form that includes several player name fields. A fixed number of players will be allowed to be registered (I suggest 8 or 16 players) (1 Mark: GUI, 2 Marks: Functionality).
    b. An **Active Tournament** will display a list of players to Anonymous users (1 Mark: GUI, 1 Mark: Functionality).
    c. Depending on the number of players there will be a fixed number of Tournament bouts initially (4 bouts for 8 players and 8 bouts for 16 players). Opponent selection for each bout will be decided in player-registration order. (2 Marks: Functionality).
4. Tournament Management **(4 Marks: GUI, 4 Marks: Functionality):**
    a. A Tournament can be **started** only after all players have been registered and the initial bouts determined (1 Mark: Functionality).
    b. A tournament will include a **fixed number of rounds** depending on how many players are registered (3 Rounds for 8 players and 4 rounds for 16 players) (1 Mark: Functionality).
    c. Each Tournament Round will be displayed as a **separate form page** (3 Marks: GUI)
    d. The **Tournament owner** will be able to select the winners for each bout in every round (1 Mark: Functionality)
    e. Only the **winners** will be carried forward into new bouts for each successive round (1 Mark: Functionality).
    f. When the final round is completed, a summary of the results will be displayed (1 Mark: GUI).

INCIDENT MANAGMENT SITE (OPTION)

**(15 Marks: GUI, 15 Marks: Functionality)**

1. User Management and site security **(5 Marks: GUI, 3 Marks: Functionality):**
    a. **User Registration** must be included. A form will allow the user to enter profile information (**username**, **password, email address, user type)** which will be stored in a database table (2 Marks: GUI, 2 Marks: Functionality).
    b. The user will be able to **Login**, **Logout** and **modify** his or her profile (1 Mark: GUI, 1 Mark: Functionality).
    c. **Site security** will prevent non-registered users from creating incident records (tickets), changing ticket status, posting a comment or seeing the incident log. (2 Marks: Functionality).
2. Incident Dashboard (Log) **(3 Marks: GUI, 4 Marks: Functionality):**
    a. After a user is **registered** and **logged in**, he or she can view an Incident **Dashboard** that will display all open Incidents (tickets) in a clickable list format (1 Mark: GUI, 2 Marks: Functionality).
    b. The Dashboard will include an option that allows the user to **create a new incident (ticket)** (1 Mark: GUI, 1 Mark: Functionality).
    c. Incidents that are closed will be initially hidden but an option on the dashboard will allow the user to view ALL incidents (1 Mark: GUI, 1 Mark: Functionality).
3. Create an Incident Record **(3 Marks: GUI, 4 Marks: Functionality):**
    a. When a new incident is created a form will be displayed that will require the user to select and/or complete several fields including: Incident Description, Incident priority, Customer information and Incident Narrative. (1 Marks: GUI, 1 Mark: Functionality).
    b. The new incident record will be stored in numerical order and the incident **record number** will be generated based on the incident date (e.g. 130418-0000001). This is typically the number provided to the customer as a reference (1 Mark: Functionality).
    c. Each Incident Record (Ticket) will include an **Incident Narrative** that will be **time-stamped** with every status change or Incident modification. This will provide detailed incident information as well as an audit trail (1 Mark: GUI, 1 Marks: Functionality).
    d. Each Incident Record (Ticket) will have a **Status Field** associated with it. Initially the Status field will be set to NEW. (1 Mark: GUI, 1 Mark: Functionality).
4. Incident Management **(4 Marks: GUI, 4 Marks: Functionality):**
    a. A registered user can change the **Status Field** of an Incident by first selecting it on the Dashboard and then selecting the appropriate Status (e.g. In Progress, Dispatched, Closed, etc.). The user **must** then enter a comment in the **Incident Narrative** (1 Mark: GUI, 1 Mark: Functionality).
    b. Once the status of an Incident Record is set to CLOSED it will not accept any further modifications. (1 Mark: GUI, 1 Mark: Functionality).
    c. Certain fields will appear **greyed-out** and will not be modifiable (e.g. Incident Record Number, Customer Name, Incident Duration, etc.) (1 Mark: GUI, 1 Mark: Functionality).
    d. Every active Incident Record will include an **Incident Resolution Field** which must be filled out before a ticket can be officially closed (1 Mark: GUI, 1 Mark: Functionality).

BLOGGING SITE (OPTION)

**(15 Marks: GUI, 15 Marks: Functionality)**

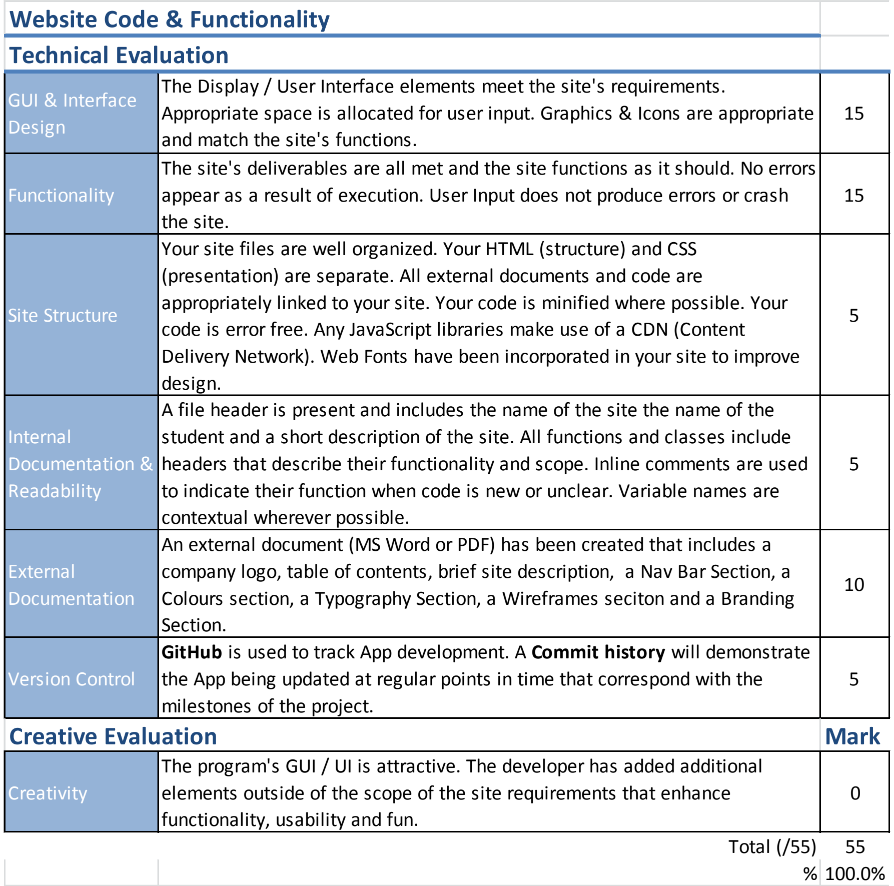
1. User Management and site security **(5 Marks: GUI, 3 Marks: Functionality):**
   a. **User Registration** must be included. A form will allow the user to enter profile information (**username**, **password, email address, etc)** which will be stored in a database table (2 Marks: GUI, 2 Marks: Functionality).
   b. The user will be able to **Login**, **Logout** and **modify** his or her profile (1 Mark: GUI, 1 Mark: Functionality).
   c. **Site security** will prevent non-registered users from creating new blogs or posting comments to an existing blog (2 Marks: Functionality).
2. Create A Blog **(6 Marks: GUI, 6 Marks: Functionality):**
   a. After a user is **registered** and **logged in**, he or she can create a **Blog.** Each Blog will present other registered users an area where they can post comments. Any number of Blogs can be created. (3 Marks: GUI, 3 Marks: Functionality).
   b. Initially, only a user's latest blog will be visible. Other blog entries will be presented in a clickable list format and ordered by date. (2 Marks: GUI, 2 Marks: Functionality).
   c. When a user creates a blog he is considered the **owner** and can choose to **close** it to any further comments. (1 Mark: GUI, 1 Mark: Functionality).
3. Users Can Post to any Active Blog **(4 Marks: GUI, 6 Marks: Functionality):**
   a. A Blog will display a **post button** to registered users that allow them to Post a comment (1 Mark: GUI, 2 Marks: Functionality).
   b. When the **post button** is selected, a registered user will **fill out a form** and will be able to add an un-moderated comment to a Blog. The form will have several **required fields** to ensure validation and security (e.g. CAPTCHA) (2 Marks: GUI, 3 Marks: Functionality)
   c. Blogs will be visible as an initial narrative entry followed by a separate list of comments from other users. (1 Mark: GUI, 1 Mark: Functionality).

## SUBMITTING YOUR WORK

Your submission should include:

1. An external document (MS Word or PDF).
2. A zip archive of your website's Project files or a link to GitHub (preferable).

Please zip all files in to a single project archive.



| Website Code & Functionality | | |
|---|---|---|
| **Technical Evaluation** | | |
| GUI & Interface Design | The Display / User Interface elements meet the site's requirements. Appropriate space is allocated for user input. Graphics & Icons are appropriate and match the site's functions. | 15 |
| Functionality | The site's deliverables are all met and the site functions as it should. No errors appear as a result of execution. User Input does not produce errors or crash the site. | 15 |
| Site Structure | Your site files are well organized. Your HTML (structure) and CSS (presentation) are separate. All external documents and code are appropriately linked to your site. Your code is minified where possible. Your code is error free. Any JavaScript libraries make use of a CDN (Content Delivery Network). Web Fonts have been incorporated in your site to improve design. | 5 |
| Internal Documentation & Readability | A file header is present and includes the name of the site the name of the student and a short description of the site. All functions and classes include headers that describe their functionality and scope. Inline comments are used to indicate their function when code is new or unclear. Variable names are contextual wherever possible. | 5 |
| External Documentation | An external document (MS Word or PDF) has been created that includes a company logo, table of contents, brief site description, a Nav Bar Section, a Colours section, a Typography Section, a Wireframes seciton and a Branding Section. | 10 |
| Version Control | **GitHub** is used to track App development. A **Commit history** will demonstrate the App being updated at regular points in time that correspond with the milestones of the project. | 5 |
| **Creative Evaluation** | | **Mark** |
| Creativity | The program's GUI / UI is attractive. The developer has added additional elements outside of the scope of the site requirements that enhance functionality, usability and fun. | 0 |
| | Total (/55) | 55 |
| | % | 100.0% |

This assignment is weighted **35%** of your total mark for this course.

All Assignments are due at the beginning of class.

Late submissions:

- 10% deducted for each day late.

External code (e.g. from the internet or other sources) can be used for student submissions within the following parameters:

1. The code source (i.e. where you got the code and who wrote it) must be cited in your internal documentation.
2. It encompasses a maximum of 10% of your code (any more will be considered cheating).
3. You must understand any code you use and include documentation (comments) around the code that explains its function.
4. You must get written approval from me via email.