# IERG4210 WEB PROGRAMMING AND SECURITY (2018 FALL)
## ASSIGNMENT MARKING GUIDELINES

## REVISION HISTORY

v1.0 Sept 7 Created this document and released version1.0

## GENERAL GUIDELINES

The assignment is designed to let students practice what they have learned in the course. Students must be aware of web application security throughout the web development. The whole assignment is split into 7 phases, leading all the way to a creative and functional shopping cart upon completion. Students should take a real-world website, parknshop.com, as a reference. In the assignment, students are expected to understand and apply proper security design principles and programming skills, regardless of which programming languages and libraries the students desire to use. The marking checklist included in the next page therefore outlines only the general requirements with a result-oriented basis in order to encourage students' creativities. For detailed guidance, students should refer to both lecture and tutorial notes.

## SUBMISSION POLICY

Students are required to package all of their source code and any external resources (e.g. database, images, css and js files) into a zip file and submit it to the course website. Each phase is associated with a firm submission deadline.

- *Early Submission Incentive* – However, for every 48-hour advanced submission in one phase, the deadline for phase 4, 5 or 6 can be extended by 24-hour, and no part thereof is accepted. For instance, submitting 100 hours earlier in phase 1 deadline will gain an extension of 48 hours for the phase 4, 5 or 6 deadline.
- *Late Submission Penalty* – Late submission will be penalized by mark deduction of $(10\%)^{1/n}$ where n is the round-up number of days delayed (e.g. 9 hrs late → 10%, 25 hrs late → ~31.6%, 49 hrs late → ~46.5%, and so forth).
- *Interim Demonstration* – Students' submissions will be randomly sampled by TAs for inspection. If a student is found unable to complete 80% of the requirements (or late submitted) in any single phase from 1 to 4, s/he is subject to the late submission penalty, and is required to give an interim demonstration (time and venue to be fixed). If the student manages to demonstrate 80% of the requirements for phase 1-4 during the demo, the penalties will be forfeited. Students capable of meeting the deadlines and requirements can neglect this interim demonstration.
- *Final Demonstration* – Students will sign up for a timeslot to demonstrate their websites to a marker, who will then grade it according to the checklist. The marker will then evaluate the student's understanding with questions.

## HONESTY IN ACADEMIC WORK

CUHK places very high importance on honesty in academic work submitted by students, and adopts a policy of *zero tolerance* on cheating in examinations and plagiarism. Students are NOT allowed to submit anything that are plagiarised. Therefore, we treat every assignment our students submit as original except for source material explicitly acknowledged. We trust that students acknowledge and are aware of University policy and regulations on honesty in academic work, and of the disciplinary guidelines and procedures applicable to breaches of such policy and regulations, as contained in the website http://www.cuhk.edu.hk/policy/academichonesty/.

# IERG4210 Web Programming and Security (2018 Fall)
# Assignment Marking Checklist v1.0

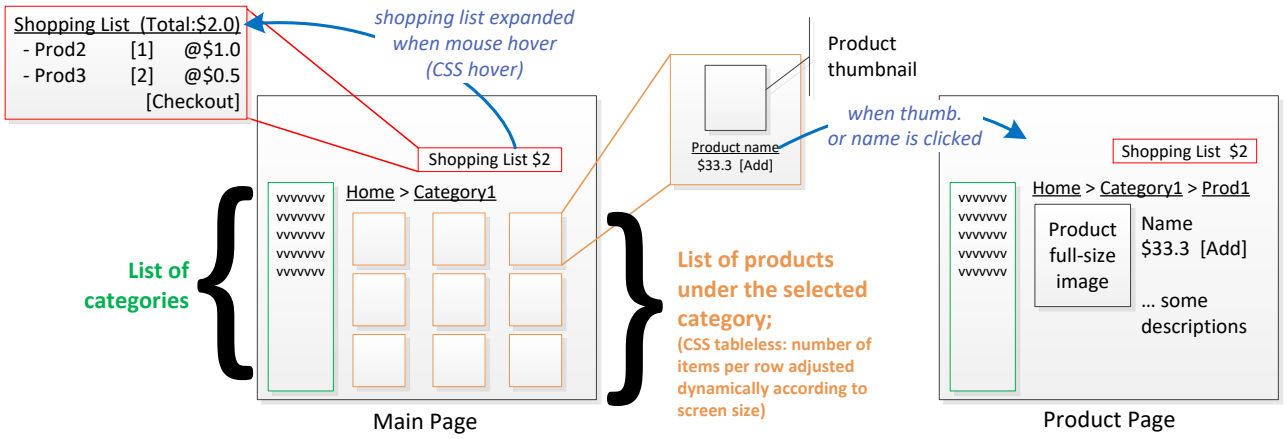
## Phase 1: Layout (deadline: sept 26, 2018, 5pm)(subtotal: 14%, bonus: 3%)

The appearance of a website plays a big role in attracting visitors. In Phase 1, you will create a **dummy** shopping website from scratch by hardcoding the basic elements. (**dummy** means categories and products are only for the purpose of displaying. Customers can not purchase goods at this moment.)

Figure 1 shows an example of a shopping website layout. **Note** that the layout you design must be different from the example we provide, at the meantime involving all the necessary features we list below. You can draw your inspiration by referencing popular shopping websites (I.e. https://parknshop.com, https://www.walmart.com).

**Declaration**:

1. Javascript is **NOT** necessary at this phase.
2. The bonus comes from the elegancy, clearness and user-friendliness of your website. The bonus depends on tutors' interpretation.




1. HTML: Make good use of semantic HTML throughout the whole assign. _____ / 2%
   - <header>, <nav>, <footer>, , <section>, <ul>, <li>…
2. CSS: Clean separation of HTML, CSS and JS code and files throughout the whole assign. _____ / 2%
   - No inline CSS and JS are allowed
   - No HTML for styling use, e.g. , align="center", etc
   - Tolerance: < 5 exceptions
3. *Main page* demonstrates the use of "CSS tableless" *product list* _____ / 2%
   - Each product has at least its own thumbnail, name, price and *addToCart* button
   - When the thumbnail or name is clicked, redirect to the corresponding product page
4. *Main page* demonstrates the use of "CSS hover" *shopping list* _____ / 3%
   - When displayed, it will cover any elements behind
   - Input boxes are used for inputting quantity of each selected product
   - A checkout button is used to submit the list to PayPal
   - The shopping list is displayed in both main and product pages
5. *Product page* provides product details _____ / 2%
   - To show a full-size or bigger image, name, description, price, and *addToCart* button
6. Both main and product pages should include a hierarchical navigation menu _____ / 3%
   - e.g. Home or Home > Category1 or Home > Category1 > Product1

- They are hyperlinks that can redirect users to an upper level of the hierarchy

## PHASE 2A: SECURE SERVER SETUP (DEADLINE: OCT 5, 2018, 5PM) (SUBTOTAL: 8%)

In this phase, you are required to setup a secure server for later development. Some guidance will be given in tutorial.

1. Instantiate a free Virtual Cloud Machine (**Amazon EC2** recommended or other free VPS) _____ / 1%
   - Details of the Free Usage Tier: http://aws.amazon.com/free/
   - You can apply for the education coupon on AWS
   - With a Linux distribution, install only Apache, PHP and SQLite (or MySQL)
     - To minimize attack surfaces, always install only what you need
2. Apply necessary security configurations _____ / 5%
   - Apply proper firewall settings at Amazon: block all ports but 22, 80 and 443 only
   - Apply proper updates for the server software packages in a regular manner
   - Hide the versions of OS, Apache and PHP in HTTP response headers
   - Do not display any PHP warnings and errors to the end users
   - Disable directory index in Apache
3. Configure the VM so that your website is accessible at http://sxx.ierg4210.ie.cuhk.edu.hk _____ / 2%
   - Apply for an elastic public IP, and ALWAYS associate it with the instantiated VM
   - Submit your elastic IP through google form link before Oct 5, 2018 5pm
   - TAs will then assign you a domain name and configure the DNS mapping for you
   - Upload all your pages to the server. They should then be accessible through:
     - http://[your-own-public-IP]/, or
     - http://sxx.ierg4210.ie.cuhk.edu.hk/

## PHASE 2B: DATA PRESENTATION & MANAGEMENT (DEADLINE: OCT 19, 2018, 5PM) (SUBTOTAL: 18%)

In this phase, you will implement the core functions of the website with mainly PHP and SQL.

1. SQL: Create a database with the following structures (to be covered in tutorial) _____ / 1%
   - A table for *categories*
     - Required columns: *catid (primary key), name*
     - Data: at least 2 categories of your choice
   - A table for *products*
     - Required columns: *pid (primary key), catid, name, price, description*
     - Data: at least 2 products for each category
2. HTML, PHP & SQL: Create an *admin panel*
   - Design several HTML forms to manage* *products* in DB _____ / 5%
     - Dropdown menu to select *catid* according to its *name*
     - Input fields for inputting *name*, *price*
     - Textarea for inputting *description*
     - ^ File field for uploading an image (format: jpg/gif/png, size: <=10MB)
   - Design several HTML forms to manage* *categories* in DB _____ / 2%

   * In terms of manage, it includes the capabilities of insert, update and delete

   ^ For the file uploaded, store it with its name based on the unique lastInsertId() _____ / 1%
3. HTML, PHP, SQL: Update the *main page* created in Phase 1
   - Populate the *category list* from DB _____ / 1%
   - Based on the category picked by user, populate the corresponding *product list* from DB _____ / 3%
     - The *catid=[x]* is reflected as a query string in the URL
4. HTML, PHP & SQL: Update the *product details page* created in Phase 1 _____ / 2%
   - Display the details of a product according to its DB record
5. Supporting automatic image resizing for product images _____/3%
   - When a large image is uploaded, the server will resize it and show a thumbnail image in panel
   - In the main page, display thumbnails. In the product description page, display the larger image.

## PHASE 3: AJAX SHOPPING LIST (DEADLINE: OCT 30, 2018, 5PM) (SUBTOTAL: 10%)

In this phase, you will implement the shopping list which allows users to shop around your products. This phase is designed to let you practice Javascript programming.

1. JS: Dynamically update[#] the *shopping list* (to be covered in tutorial)
   - When the *addToCart* button of a product is clicked, add it to the shopping list _____ / 1%
     - Adding the same product twice will display only one row of record
   - Once a product is added,
     - Users are allowed to update its *quantity* and delete it with a number input, or two buttons for increment and decrement _____ / 1%
     - Store its *pid* and *quantity* in the browser's `localStorage` _____ / 2%
     - Get the *name* and *price* over AJAX (with *pid* as input) _____ / 3%
     - Calculate and display the total amount at the client-side _____ / 1%
   - Once the page is reloaded, the *shopping list* is restored _____ / 2%
     - Page reloads when users browse another category or visit the product detail page
     - Populate and retrieve the stored products from the `localStorage`

   [#] The whole process of *shopping list* management must be done without a page load

## PHASE 4: SECURING THE WEBSITE (DEADLINE: NOV 20, 2018, 5PM) (SUBTOTAL: 30% (33% FOR MSC STUDENTS))

In this phase, you will protect your website against many popular web application security threats.

1. No XSS Injection and Parameter Tampering Vulnerabilities in the whole website
   - [UI Enhancement Only] Proper and vigorous client-side input restrictions for all forms _____ / 1%
   - Proper and vigorous server-side input sanitizations and validations for all forms _____ / 2%
   - Proper and vigorous context-dependent output sanitizations _____ / 2%
2. Mitigate SQL Injection Vulnerabilities in the whole website _____ / 2%
   - Apply parameterized SQL statements with the `PDO` library
3. Mitigate CSRF Vulnerabilities in the whole website _____ / 2%
   - Apply and validate secret nonces for every form
   - ALL forms must defend against Traditional and Login CSRF
4. Authentication for Admin Panel (sample code and details to be given in tutorial)
   - Create a user table (or a separate DB with only one user table) _____ / 1%
     - Required columns: *userid (primary key), email, password*
     - Data: *at least 2 users of your choice, 1 admin and 1 normal user (using admin flag)*
     - Security: Passwords must be properly salted and hashed before storage
   - Build a *login page* `login.php` that requests for *email* and *password* _____ / 3%
     - Upon validated and authenticated, redirect the user to the *admin panel* or main page
     - Indicate user name (or "guest" if not logged in) in your website
     - Otherwise, prompt for errors (i.e. either email or password is incorrect)
     - A separated normal user login page is not compulsory
   - Maintain an authentication token using `Cookies` (with `httpOnly`)
     - Cookie name: `auth`; value: a hashed token; property: `httpOnly` _____ / 2%
     - Cookies persist after browser restart (i.e. 0 < expires < 3 days) _____ / 1%
     - No Session Fixation Vulnerabilities (rotate session id upon successful login) _____ / 1%
     - Configure all authentication cookies to use the Secure and HttpOnly flags _____/ 1%
   - Validate the authentication token before revealing and executing admin features _____ / 3%
     - If successful, let admin users access the admin panel and execute admin features
     - Otherwise (e.g. empty or tampered token), redirect back to the *login page* or main page
     - Security: both admin.html and admin-process.php must validate the auth. token
   - PHP & SQL: Provide a logout feature that clears the authentication token _____ / 1%
   - Supporting Change of Password _____/ 2%

        - Must validate the current password first
        - Logout user after the password is changed
5. **(For MsC. Students only)** Supporting member management for buyers _____ / 3%
    - Create a member portal for buyers – sign up, sign in, sign out and change password.
6. All generated session IDs and nonces are not guessable throughout the whole assign. _____ / 1%
    - e.g., the login token must not reveal the original password in plaintext
    - e.g., the CSRF nonce when applied in a hidden field must be random
7. Apply SSL certificate for `secure.s[1-80].ierg4210.ie.cuhk.edu.hk` (to be covered in tutorial )
    - Certificate Application _____ / 2%
        - When generating a CSR, use `CUHK` as Organization Name
        - Apply a 90-day free certificate at https://www.ssl.com/certificates/free/buy/ (or others)
        - Reminder: the application process can take more than a day, so apply early!!
    - Certificate Installation
        - Install the issued certificate and apply security configurations in Apache _____ / 1%
            - Apply strong algorithms and secure cipher suites
        - Host admin panel at https://s[1-80].ierg4210.ie.cuhk.edu.hk/admin.php _____ / 2%
            - In the .htaccess ( other ways are also OK ), redirect users to https website if come from:
              `http://[secure...]` or `http://[...]/admin.php`

## PHASE 5: SECURE CHECKOUT FLOW (DEADLINE: DEC 2, 2018, 5PM) (SUBTOTAL: 16% (28% FOR MSC STUDENTS))

This is a tough phase, yet the most critical phase to escalate the professional level of your website to the next level. (You'll likely be offered a job if you can demonstrate such a level of web programming skills) The implementation has already been outlined as below. Be prepared to spend substantial amount of time in debugging.

1. Sign up at https://developer.paypal.com/ and create two test accounts: _____ / 1%
    - A merchant account - after logging in to the Sandbox Test Site, modify necessary settings in the Selling Preferences under Profile
    - A buyer account – use it to pay for purchased items in your shopping portal
2. Enclose your shopping cart with a <form> element _____ / 3%
    - Use the Cart Upload Command of PayPal Website Payment Standard (cmd=_cart&upload=1)
    - Insert additional hidden fields that are required by PayPal (Read the first reference)
        - business, charset, currency_code, item_name_X, item_number_X, quantityX
        - invoice and custom
    - Create a checkout button that submits the form
3. When the checkout button is clicked: _____ / 4%
    - Pass ONLY the *pid* and *quantity* of every individual product to your server using AJAX and cancel the default form submission
    - Server generates a digest that is composed of at least:
        - Currency
        - Merchant's email address
        - A random salt
        - The *pid* and *quantity* of each selected product (Is quantity positive number?)
        - The current price of each selected product gathered from DB
        - **The total price of all selected products**

      *Hint: separate them with a delimiter before passing to a hash function*
    - Server stores all the items to generate the digest into a new database table called *orders*
        - The user could be logged in or as "guest" to purchase, store username with order in DB

- Pass the lastInsertId() and the generated digest back to the client by putting them into the hidden fields of invoice and custom respectively
- Clear the shopping cart at the client-side
- Submit the form now to PayPal using programmatic form submission

4. Setup a Instant Payment Notification (IPN) page to get notified once a payment is completed
   - Validate the authenticity of data by verifying that it is indeed sent from PayPal _____ / 1%
     - Your IPN receiver page is served over HTTPS (using the SSL cert)
     - When contacting PayPal for message authenticity check, use SSL and port 443
     - The sample code of validation protocol will be given in tutorial 9

     *Hint: sample code will be given in tutorial*
   - Check that txn_id has not been previously processed and txn_type is cart _____ / 1%
   - Regenerate a digest with the data provided by PayPal (same order and algorithm) _____ / 2%
   - Validate the digest against the one stored in the database table *orders* _____ / 2%
     - If validated, the integrity of the hashed fields is assured
     - Save the txn_id and product list (pid, quantity and price) into DB

   *Debugging Hint: use error_log(print_r($_POST,true)) to print out the parameters passed by PayPal*
5. After the buyer has finished paying with PayPal, auto redirect the buyer back to your shop _____ / 1%
6. Display the DB *orders* table in admin panel: product list, payment status…etc. _____ / 1%
7. **(For MsC students, refer to Phase 4, entry 5.)** Let members check what they have purchased in the most recent five orders _____/2%
8. **(For MsC students)** Apart from Paypal, many corporations provide other payment approaches to facilitate the purchase, like Alipay, Wechat Pay and so on. In this step, you will add a second payment approach to your website. _____/ 10%
   - You can choose any payment approach except Paypal.
   - Your new payment approach should provide the same functions as Paypal.

Client-side Demonstration: ----- References:
https://developer.paypal.com/docs/classic/lifecycle/sb_overview/
https://developer.paypal.com/docs/classic/paypal-payments-standard/integration-guide/button_summary/
https://developer.paypal.com/docs/classic/ipn/integration-guide/IPNIntro/
http://www.evoluted.net/thinktank/web-development/paypal-php-integration

## PHASE 6: EXTENSIONS (DEADLINE: BEFORE DEMO) (SUBTOTAL: 9%, BONUS: 7% MAX)

In this phase, you can choose any combinations of the following items to implement. At most 7% bonus will be awarded.

1. Mashup: Including a social plugin in the main page _____ / 1%
   - Facebook: https://developers.facebook.com/docs/plugins/
2. SEO: Apply search engine optimized (or user-friendly) URLs when browsing products _____ / 2%
   - Include the name of categories and products into the URLs:
     e.g. http://s0.ierg4210.ie.cuhk.edu.hk/2-Fruits/ for browsing products under the category Fruits
     e.g. http://s0.ierg4210.ie.cuhk.edu.hk/2-Fruits/9-Apple for browsing product details
   - You can map the above URLs to your php using apache scripts (Hint: google RedirectCond)
3. Supporting pagination/AJAX infinite scroll when browsing products in the main page _____ / 3%
4. Supporting HTML5 Drag-and-drop file selection in the admin panel _____ / 2%
   - Create a dropping area that takes an image
   - Display a thumbnail (i.e. smaller width and height) if the dropped file is an image; reject it otherwise
5. Supporting multi-level categories for products (should update both frontend and backend) _____ / 3%
6. Using AJAX when browsing categories and products in the main page _____ / 3%
7. Making use of additional services provided by AWS (Note: **charges may apply!**)
   - Making use of the SES email services when sending emails, if any _____ / 4%
   - Let admin upload images directly to S3 Storage, and serve the files from there _____ / 5%
8. Support "upload file from Dropbox" function in admin panel _____ / 5%
9. Supporting multi-session management _____ / 6%
   - Show the simultaneous logged-in sessions in the admin panel
   - Each session should be identified by an IP and allows logging out other sessions
   - Hints: Use DB to save valid authentication token. Examples: Gmail and Dropbox
10. Supporting the use of gift vouchers (e.g. EASTER12 for $5 discount) _____ / 6%
    - Create a DB table called vouchers that store voucher code and the corresponding discount
    - Add a field for voucher code just above the checkout button
      - Auto fill the voucher code if it is supplied through a query parameter (e.g. ?vcode=EASTER12)
    - Use AJAX to dynamically validate the coupon code (using onkeydown/onblur handler)
    - Apply discount and update the UI to reflect the discounted price and the discount amount
    - Security: Make proper validations throughout the checkout process
    - Hints: HTML variables in p. 433 of the first reference
11. Mashup: Supporting Secure Authentication with Google or Facebook accounts _____ / 6%
    - Google: http://code.google.com/apis/accounts/docs/OAuth2.html
    - Facebook: https://developers.facebook.com/docs/authentication/
12. Supporting secure password reset through email _____ / 6%
    - A page that asks for email address for password recovery
    - Only if the email corresponds to an existing user, an email will be generated
    - In the email, a password recovery hyperlink will make use of a random nonce
    - Only the admin receiving the nonce can reset his/her password
13. **(For undergraduate students)** Supporting member management for buyers _____ / 5%
    - Create a member portal for buyers – sign up, sign in and sign out
    - Let members check what they have purchased in the most recent five orders
14. Supporting discounts when purchasing multiple quantities of a product type _____ / 8%
    - Create a DB table called discounts that store conditions for applying discounts
      - Conditions could include: "buy 2 get 1" and "buy $10@2, $6@1"
      - Refer to parknshop.com for reference and details
    - Update the UI to reflect the discounted price whenever the quantity conditions are met
    - Security: Make proper validations throughout the checkout process

- Hints: HTML variables in p. 433 of the first reference

15. TBD – May release more options upon students' requests

## PHASE 7: PEER HACKING (TBD) (SUBTOTAL: 15%, BONUS: 5% MAX)

It is critical to defend against potential attacks *before* they turn into reality when the website is open to the public. Students are to exercise *ethical hacking* in this phase. Be reminded to backup everything (code, conf files).

1. Practice with the use of any automated vulnerability scanner _____ / 0%
   - Use an automated vulnerability scanner (e.g., Nikto, Skipfish or others). Fix your vulnerabilities, if any.
   - Scan ONLY your own website to see what common errors you could have made
   - Note: Beware of exceeding the bandwidth quota so that you are charged

This is a game to help you learn security practically, which starts on Dec 4 and ends on Dec 8. Site owners should responsibly react to the incidents and fix the problems as soon as possible to mitigate future exploits. Each student is entitled a base score of 10, while the highest score is 21. Students must report any bug issues at Google Form:

1. Perform manual ethical hacking for shops of your classmates _____ / 16%(21%)
   - For a vulnerability in Student V's shop being reported by Student R1: R1 + 4%
     - Student R1 reports and clearly specifies the problems: V-3%
     - **Only the first 5 entries a student submits will be counted**. We will sort your records based on the time you submit. Cherish your chances and make sure the bug really exists.
     - When reporting, Student R1 MUST strictly follow the following format
       - Shop Name with vulnerability: e.g. s81
       - You SID/shop ID
       - Type of Vulnerability: Prefixed with OWASP Code, e.g. [A3] XSS
       - Content: Answer 4 Questions in the report link
       - Attach at least one screen capture(s) for a proof and to aid illustration
     - If a bug report is invalid:
       - To dispute a bug, student V must load his/her source code from eLearn and demonstrate it to TA during final demo
     - Student V fixes the problem on or before FINAL DEMO
       - Should point out the solutions in demo
       - **No marks** will be deducted
   - For a non-security related bug in Student V's shop being reported by Student R:
     *Please understand that some students might skip doing a part or two, regarding them as bugs to report are meaningless to the aim of this phase. So, unless your discovery is a kind of system flaw, otherwise, do not report it.*
     - The marks allocation is half of the security ones: R1 + 1%
     - Each bug MUST be prefixed by the numbering of assignment requirements V -1%
       - Example number: [P4-1] stands for requirement 1 in phase 4
     - The bug must be reproducible in both Firefox and Chrome
     - Each student can report at most 3 non-security related bugs
   - Availability check by the automated IERG4210-robot
     - http://s68.ierg4210.ie.cuhk.edu.hk will report availability of shops during whole period
     - Deduct 2 mark if not functioning for 1 day: V -2% / day
     - Also do not hide your website page intentionally
   - Student R causes high volume of traffic on others' websites
     - Includes but not limited to launching automated scan, DoS, or DDoS
     - Please shutdown the service to avoid being charged and submit the access_log to us ASAP

This game is expected to be quite exciting! :) This is an internal ethical hacking exercise, hence, students whose websites are exploited or found vulnerable should not take it as offensive but a good learning opportunities. Students should

respect each other and be polite in any circumstances. In case of dispute, TA will be the final judge on the marking. Drastic circumstances (e.g. bullying) can result in penalties when the instructor deems it appropriate

FINAL Q&A (SUBTOTAL: -50%)

- Random Question 1: Code-related _____%
- Random Question 2: Concept-related or Code-related _____%

SID: _____

TOTAL: _____ / 120% (+15% BONUS)
135% (+15% BONUS) FOR MSC STUDENTS
MARKER RESPONSIBLE: _____