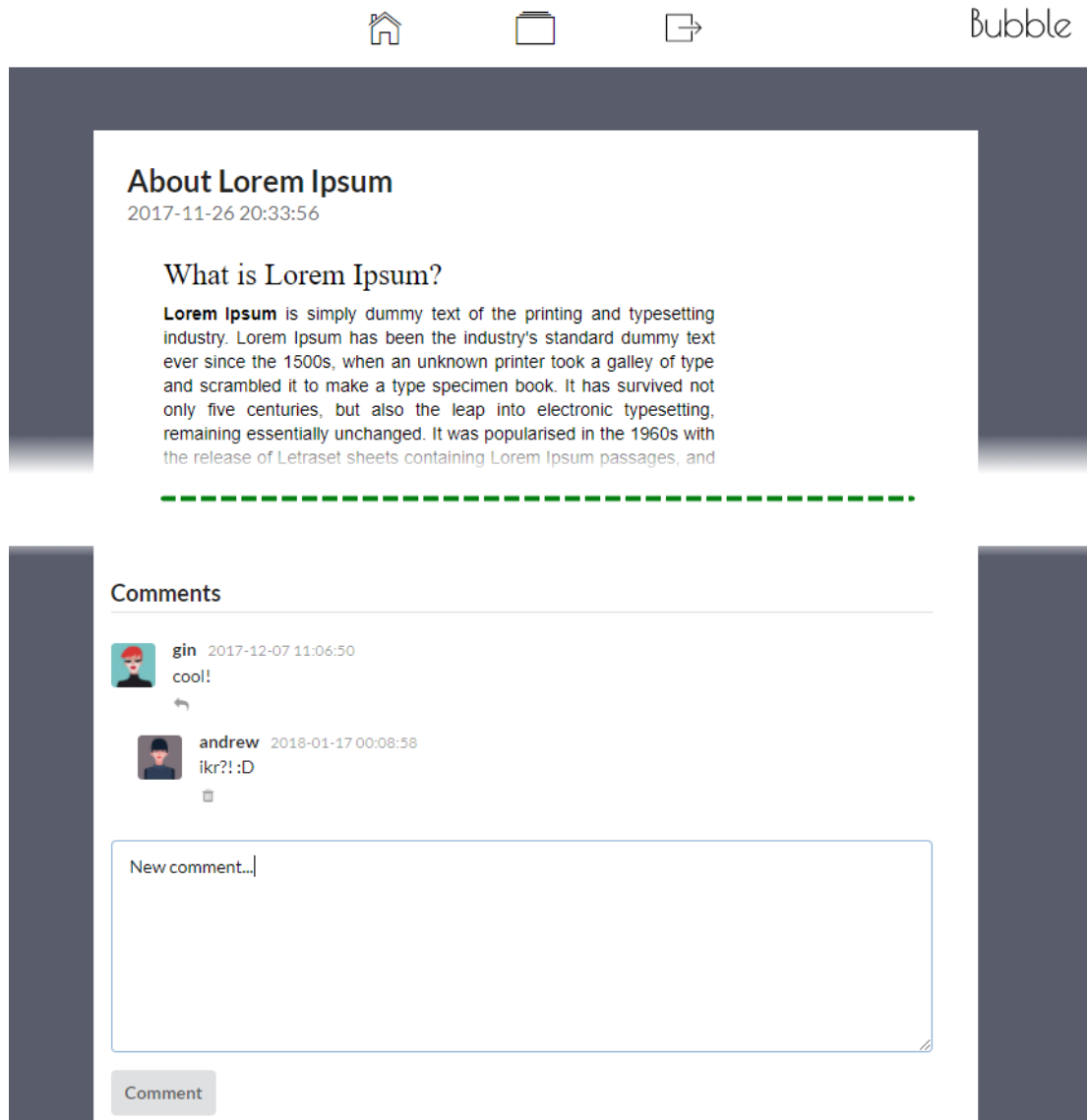




PGCert in Information Technology

Final Project - A Personal Blogging System



Credit: "Bubble Blog" (Emma Zhao, Tongxin Xie, Qian Peng, Xingjian Che)

Important Dates

- **Thursday May 17th:** Project introduction
- **Tuesday June 5th:** Presentation day
- **Wednesday June 6th:** Source code submission
- **Friday June 8th:** Final report submission

Introduction

In this assignment you will develop a blogging website using a combination of the programming skills you have learnt through the *Programming for Industry* and *Programming with Web Technology* courses. The snapshot shown above is taken from a previous PGCert IT student group's submission.

Through the web site, users can register for an account, which is needed to be able to post articles and to leave comments on others. When logged in, they have full control of the content they have authored: creating, updating and deleting their content and comments.

The functionality you need to provide to implement the blogging system, which is detailed below, has been divided into two sections: basic functionality, which is compulsory, and advanced features. For advanced features, there are a range of features to choose from—you do not have to implement them all.

This project is a team project - each of you will be working in groups which will be assigned by your lecturer. Make sure to name your team something memorable!

Team Deliverables (worth 20% overall)

The team portion of this project consists of **100 marks total**, and is worth **20%** of your final grade for both the *Industry* and *Web* papers in the certificate. The marks are comprised as follows:

Deliverable	Marks
Implementation of compulsory features	32 marks
Implementation of advanced features	48 marks
Code style	10 marks
Daily stand-ups	10 marks
Presentation & demonstration	0 marks (but compulsory)

Details of the individual deliverables are given in these subsections.

Implementation of compulsory features (32 marks)

There are a set of compulsory features which comprise the basic functionality for a blogging system. These are detailed in the “compulsory features” section below. Your team **must** implement **all** of these features adequately to receive full marks for this section.

Implementation of advanced features (48 marks)

There are a wide range of additional features which could be implemented to improve the functionality and user experience of your team's blogging system. These are organized into **feature packs** which are detailed in the "advanced features" section below. Your team must select and implement **three** of these feature packs, for **16 marks each**.

Code style (10 marks)

Your code must be easily understandable by third parties, and conform to best practices. This includes use of appropriate variable names, sufficient commenting, and adherence to applicable patterns such as DAO and Web-MVC, amongst other considerations.

Daily stand-ups (10 marks)

Each weekday during the project (other than public holidays), your team will be required to report progress to the lecturer. Each team member must be present at every meeting. Further details are available in the *Project Management* section below. The meetings will be short - approximately five minutes per team per meeting. Evidence of good teamwork is required at these meetings - your scrum master should be able to see and agree upon a fair workload allocation for each team member. Evidence of good project planning should also be visible in these stand-ups, via a team [Trello](#) board.

Presentation & demonstration

On **Tuesday, June 5th**, each team will get the opportunity to present their project to the rest of the class, and other staff and guests. This is your chance to show off all your hard work and should be a fun session. The format of this presentation is a secret - you will find out the exact details on the day!

Following the presentations, all attendees will get the opportunity to try out each team's project. This means that, while the source code is technically due on the 6th, it is highly recommended to be functionally complete before the 5th.

Individual Reflective Report (worth 10% overall)

In addition to the team component, you should submit an **individual** reflective report (each team member must write their own), which comprises **10%** of your final grade for both the *Industry* and *Web* sections of the course. This is due on **Friday, June 8th**. The report should cover the following topics:

- In your own words, explain how the system as a whole has been designed;
- Detail your particular contributions to the team and project;

- Detail which topics, taught in class, have been used within the project, and where;
- Detail any topics or technologies, which were not taught in class, that you have used, and where;
- Describe the lessons you have learned from working in a team. What are the benefits, and are there any drawbacks or difficulties? How have those been overcome in your team?

The length of the report should be approximately **four – six pages**, or around 1500 - 2000 words.

Compulsory Features

Your team must implement the following compulsory features, which comprise the basic functionality of a blogging system:

1. Users must be able to create new accounts. Each new user should be able to choose a username (which must be unique) and a password. At minimum, a user's real name, date of birth, and country should also be recorded, along with a brief description about themselves.
2. When creating an account, users must be able to choose from amongst a set of predefined "avatar" icons to represent themselves. Users must also be able to upload an image from their computer, from which a thumbnail can be generated for use as a custom avatar.
3. Once a user has created an account, they must be able to log in and log out.
4. Users must be able to browse a list of all articles, regardless of whether they are logged in or not. If logged in, they should additionally be able to browse a list of their own articles.
5. When logged in, users must be able to add new articles, and edit or delete existing articles.
6. When logged in, users must be able to comment on articles. Users must also be able to delete any comments they have written, as well as any comments on articles which they have authored.
7. Users must be able to edit any of their account information, and also be able to delete their account.
8. The website must have a consistent look and feel, and must be responsive.

Using default Bootstrap, w3.css or similar is ok, but do not use any commercial or freely available 'website templates'. If in doubt, *check with your lecturer first*.

Advanced Features

Your team must choose and implement **three** of the following six feature packs detailed in this section to improve the functionality and user experience of your blogging system.

Pack One: Administrative interface

An administrative interface should be implemented using Swing, JavaFX or Web technologies that allows an administrative user to:

1. Add and remove users
2. Reset user passwords, by sending the corresponding user an email with a link to reset their password
3. Show and hide comments and articles

Pack Two: User interface

Your blogging system should provide the following improvements to the website's user experience:

1. WYSIWYG support for authoring articles
2. Informing new users whether or not their chosen username is already taken - *without* the user having to submit a form to find out (i.e. use AJAX).
3. The website should be styled using a distinctive theme, going above and beyond default bootstrap / W3.css to give your website a clean, professional and unique look.

Pack Three: Feature enhancements

Your blogging system must provide the following extra features to users:

1. When creating new accounts, users should be verified as human using a Google's reCAPTCHA.
2. Users must be able to enter a date when authoring an article. The date should default to the current day, but the user can change this to any valid date (past or future) they wish. If the user enters a future date, the article shouldn't be visible until that date.
3. Users should be able to **search** and **sort** article lists by *article title*, *username*, and *date*.

4. Nested comments should be supported - i.e. users should be able to comment on articles, *and* on other comments.

Pack Four: Multimedia

Your website should incorporate the following support for multimedia:

1. Users should be able to upload any number of images to an article. Multiple images per article must be supported, and these images should be able to be added and removed when editing existing articles, as well as when authoring new ones. Ideally, if you're also implementing Pack Two (see above), then your WYSIWYG article editor should allow images to be uploaded anywhere within the article's content (though this is not a strict requirement).
2. Users should be able to upload audio and video files to an article. Support for just one audio / video item per article is ok.
3. Users should be able to embed YouTube videos within an article. If implementing a WYSIWYG editor, these embedded videos should ideally be able to be placed anywhere within the article's content.
4. Users should be able to browse a multimedia gallery showing all multimedia content that's been uploaded to any article. Users should be able to switch between looking at just their own multimedia, or content from all users. When not logged in, users should still be able to browse all content.

Pack Five: Robustness & security

Your website should implement the following features to make it more resilient to attacks and unexpected user input:

1. Passwords should not be stored in plaintext. They must be appropriately hashed / salted. Alternatively, you can investigate and use a 3rd-party authentication system (SSO), such as "log in with Facebook", or "log in with Google", etc.
2. You should comprehensively check all input, and provide appropriate non-default error responses when required. What happens when users browse to non-existent pages? What happens when users bypass your HTML5 / JavaScript form validation and submit incorrect values (this is very easy to do with appropriate tools)? What happens if a malicious user attempts SQL injection? These questions and more should be considered!
3. Any other items / checks from the OWASP Top 10 which may be applicable.

Pack Six: Roll your own pack

Do you have some amazing features you want to implement, which aren't included in the other five packs? If these features are sufficiently impressive and complex, you can make your own feature pack from them. The sky's the limit - but you must **consult your lecturer** to make sure your desired features will meet the requirements.

Project Management

There are several project management considerations to keep in mind while working on this project.

Teamwork

With the exception of the reflective report, the rest of the project is a team project. You will be expected to work with members of your team to effectively come up with a list of requirements, prioritize that list, and divide the work up amongst team members. While working on the project, also keep in mind that each team member will need to demonstrate that they have worked on both the front-end (HTML / CSS / JavaScript) and back-end (Servlets / JSPs / MySQL), and that they have contributed equally and fairly to the team. Those considerations notwithstanding, how you divide the work amongst your teammates is entirely up to you, but **your scrum master must agree on your team's proposed work breakdown**. Your team will lose marks if there is evidence of poor teamwork or unfair task allocation.

Pair Programming

It is *compulsory* to do pair programming for this project. Each team member must demonstrably spend some of their time pair programming, both as the driver and as the navigator. In a group with three people, having one pair and one individual will be far more effective than trying to split the work three ways.

Communication

Remember to organize some easy way in which your group can communicate – especially if you're not working in the same location (though working in the provided lab space for the majority of the time is strongly encouraged). Facebook works well for this, but if you don't want the distraction, we recommend trying Slack. Or, feel free to use any other tool your group agrees upon.

Code Sharing

It is vital to properly utilize version control to reduce headaches when working on the same codebase. The first thing your team should do is create a BitBucket repository, which you'll all use collaboratively. A video from the University of Waikato is available [here](#), which explains a little more about how to use git collaboratively in an effective manner.

Demonstrating Progress

You will be required to keep an up-to-date visualization of your team's progress at all times. To do this, your team should set up a [Trello](#) board. This free online tool will allow your team to easily keep track of tasks, add them to different lists such as "To-Do", "Doing", "QA", and "Done", assign tasks to team members, and color-code the tasks for ease of viewing. You will also add your scrum master to your board so they may keep track of your progress easily.

During the regularly-scheduled class times, you will be required to make a short progress report to your Scrum master (one of the lecturers / tutors). This will involve a stand-up session where your team will briefly report on what you've done, what you're currently doing, and what you still need to do later.

Note: Keeping track of progress this way should not just be considered "busy-work" – it serves a real purpose, in that your team will be able to set clear goals and keep them in sight at all times!