Major Project: Electronic Billboard Display and Management System

CAB302 Semester 1, 2020

Date due: 31/05/20 (Sunday, Week 13)

Weighting: 60%

Group size: 3-5

Specification version: 1.0

Overview

The unnamed client corporation has contracted your team of 3-5 developers to create a system for handling electronic billboards. The situation is that they have a bunch of screens placed around their various offices all over the country, all of which are hooked up to computers, which they use to show announcements, advertising, motivational images etc. Their current approach to managing the content on these displays requires a large amount of manual work and the corporation has contracted your team to design a replacement that fits their specific needs.

The setup this corporation wants is three software applications, all connected via a network. One application, henceforth referred to as the **Billboard Viewer**, is a GUI application that will actually display the contents, filling the screen. It will connect to the **Billboard Server**, which acts as the central control hub for all billboard viewers connected to it. The server will *not* be a GUI application and users will not interact with it directly. Instead, users will run a second client program, the **Billboard Control Panel**. This will be a GUI application that will connect to the server and allow the user to carry out various management tasks, such as changing which billboards are shown at which times. The server will store its information in a MariaDB database.

In this scenario we have collected a number of user stories from project stakeholders, which you will use as your reference for the design work. There are explicit design requirements described in this document, but where requirements are not explicitly stated, the design decisions are left up to you to implement, using these user stories as a reference.

User Stories

"I will be running the Billboard Viewer program on the displays, and I want it to be a program I can just run and then not have to do anything else with."

"I'm a manager and I want to control which employees have access to which features. I should be able to grant the following combinations of privileges to Billboard Control Panel users:
- Creating new billboards
- Editing existing billboards
- Scheduling billboards to be displayed on the Billboard Viewers
- Editing users (incl. creating new users and editing user permissions.)
Note that, if you created a billboard, you should be able to edit that without any special permission. The 'Editing existing billboards' permission is just for editing others' billboards."

"I'm a billboard designer. We get a lot of requests, but most of the designs are pretty simple. We typically have some nice big text at the top, a picture in the middle, and some more information down at the bottom. Sometimes we only have one or two of those elements, but you get the idea. I want to be able to just type the text in, pick a picture, set some colours and go."

"I'm a billboard scheduler. I need to be able to see the billboards that are available on the system, see what they look like and then schedule them to run at particular times. Sometimes I might need to show some important information immediately. At other times I might want to set a billboard to show, say, at 6pm every day. I need to be able to specify a time as well as how long the billboard is to be shown for and how often it is to be shown."

"I'm the security lead and I care a lot about password security. User passwords should be stored as a salted hash in the database (with a unique salt generated for each user), so that the plaintext password is never sent over a network nor stored in any server."

"I'm a user and I want to be able to change my password every now and then. Users should be able to change their own passwords from the Control Panel without needing special privileges."

"I'm a manager, and I know that billboard creators need to be able to edit and delete their own billboards, but can you make it so that this doesn't apply to billboards that have already been scheduled? Once a billboard scheduler has given a billboard the all-clear and scheduled it, it should be locked. Obviously this doesn't apply to those power users who can edit and delete any billboards; I'm more worried about the billboard creators."

"I'm in charge of technical services and the Viewer, Server and Control Panel should all be network-configurable via a properties file:
    -      For the Viewer and Control Panel, containing the address and port of the server.
    -      For the Server, containing the port to listen on.
This will allow my team to rapidly reconfigure in case we have to move the server to a new machine."

"I'm the database server administrator. The Billboard Server should be configurable through a **db.props** properties file *(note: this file is described later on in this specification)*, containing fields describing how to connect to the database server and the database connection to use. This will allow us to change where the database is hosted and even change the type of database in use without recompiling the application."

Explicit Requirements

- You are to create a set of three Java applications (these can all be in one project or in three projects, whichever works for you.
- Application #1 is the Billboard Viewer, described in more detail below. It will connect over a network (LAN or internet) to the Billboard Server and display whatever the server tells it to, without requiring further user involvement. This will be a GUI application using Java Swing.
- Application #2 is the Billboard Server, also described in more detail below. The Billboard Viewer and Billboard Administrator will connect to it over LAN or internet. It will connect to a MariaDB database via JDBC. The database will store information about the contents of billboards, their scheduling, and user information used by the billboard administrator. This will be a command-line application that will run on a server and will not be interacted with directly by users.
- Application #3 is the Billboard Control Panel, also described in more detail below. This will be another client-facing GUI application using Java Swing, but it will be used to configure the billboard system— adding new billboard designs and configuring when they are displayed.
- You will use the Java JDK 13 to develop these applications. Your assignments will be tested on OpenJDK 13.
- You may use whatever development environment you wish to develop this project-however, your assignment will be marked from within IntelliJ IDEA, to test things like unit test code coverage. This means you must submit a project that we can load in IntelliJ IDEA, irrespective of what development environment you actually used.
- You will use JUnit 5 Jupiter to write unit tests for your applications. You will need to create mock objects in places to test components that rely on communicating to
- Aside from the Java JDK 13 and JUnit 5, **you are not permitted to use any external libraries.** The JDK has a very extensive API and everything you need for this assignment can be found in there, so this should not cause you any problems.
- You are to use a Git to version control the entire project (or projects). Each member of your team's individual contributions should be committed to the repository under that individual's name, to ensure everyone gets credit for the work they did.
- If you make use of a remote hosting service for Git projects (and it is highly recommended that you do), such as GitHub or BitBucket, ensure that your repository is **private** so that your hard work cannot be stolen by others.
- You are allowed to make use of tools like IntelliJ's GUI Designer to create your GUI, as long as external libraries are not used.

Billboard Viewer

- The Billboard Viewer is a GUI application, but it does not exactly have much of an interface. Its job is to simply use Swing to display billboards full-screened, updating their contents as the billboards change.
- Upon starting up, and every fifteen seconds after, the Billboard Viewer connects to the Billboard Server and receive the current billboard to be displayed.
- The network details required to connect to the server will be read out of a Properties file. The exact format of the Properties file is up to your team; however, it should be identical to the one used by the Billboard Control Panel (so the same file can configure both programs).
- If the Billboard Server is not available when the Viewer attempts to retrieve the latest billboard, it should display an error screen instead of a billboard. The design of the error

screen is up to your team. It should **not** present any kind of dialog box that requires human interaction, and it should continue requesting updates every fifteen seconds so that when the server comes back up the billboards continue displaying as normal.

- Pressing the 'Escape' key on a keyboard or clicking the mouse anywhere on the billboard screen closes the billboard viewer.
- The Billboard Viewer should be a true fullscreened application. While running there should be nothing else visible on the screen except the contents of the currently displayed billboard- no window borders or anything. (On a multi-monitor setup the billboard is only expected to fill one monitor.)
- The details describing how billboards are stored and transmitted appear later on in this document.

Billboard Control Panel

- The Billboard Control Panel is another GUI application that connects to the BillBoard Server.
- It has the following main purposes:
  - Creating new billboards to be displayed on the BillBoard Viewers (note: the Billboard Viewer and Billboard Control Panel **never directly communicate.** The Billboard Control Panel uploads billboards to the Server and the Viewer downloads them from the Server.)
  - Modifying existing billboards.
  - Scheduling billboards to be displayed at particular times.
  - Maintaining the Server's list of authorised users and user permissions.
- Upon starting up the application, the Billboard Control Panel will display a login screen, prompting the user to enter a username and password to proceed. There is no way to access the system without entering in a valid username and password. More information about user authentication is given later in this document.
- After successfully logging in, the rest of the UI design is up to your team, but you will need to support several pieces of functionality described below:
  - Create Billboards
    - Users with the "Create Billboards" permission can create new billboards.
    - Each Billboard has a name, which will be entered during creation.
    - The Billboard XML format is described later in this document. Users can export and import a billboard .xml file in this format.
    - Users will be presented with a user-friendly interface for designing the billboard. Once again, the design of this is up to your team. This will involve setting text to appear on the billboard, choosing colours for the text and background, adding an image etc.
    - Users will be able to access a graphical preview of the billboard, to see how it will look without having to upload it to the Server and view it in the Viewer.
  - List Billboards
    - All users will be able to access a list of all billboards on the system and preview their contents.
    - Users with the "Create Billboards" permission will also be able to edit or delete any billboards they created, as long as those billboards are not presently scheduled.

- Users with the "Edit All Billboards" permission will be able to edit or delete any billboard on the system, including billboards that are currently scheduled.
- The interface for editing a billboard should be the same as the interface for creating one, except autopopulated with that billboard's existing contents and formatting.
  - o Schedule Billboards
    - Users with the "Schedule Billboards" permission will be able to schedule billboards to be displayed on the Viewers.
    - Users will be presented with a calendar view, showing billboards scheduled for the week.
    - Users will be able to schedule a billboard to show at any time (to the day, hour and minute), for any duration (in minutes).
    - When scheduling a billboard, the user may specify that the showing is to recur every day, every hour or every _____ minutes (with a limitation that a billboard may not be scheduled more frequently than the duration of the billboard.)
  - o Edit Users
    - Users with the "Edit Users" permission (administrators) will be able to access a list of all users and can both edit any user listed in the system as well as create new users.
    - When creating a new user, the administrator creating the user will have to enter in a username and password, as well as select which permissions the user will have, out of the following list:
      - "Create Billboards"
      - "Edit All Billboards"
      - "Schedule Billboards"
      - "Edit Users"
    - Administrators will be able to modify existing users. This includes changing an existing user's password and changing an existing user's permission.
    - Administrators will not be able to remove their own "Edit Users" permission- however they can remove the "Edit Users" permission from other administrators.
    - Administrators will also be able to delete users- however, as a security precaution, administrators will not be able to delete their own accounts. Administrators can delete the accounts of other administrators, however.
    - Note that **all users** will be able to change their own passwords, though only users with "Edit Users" permission can change the passwords of other users.

Billboard Server

- The Billboard Server does not have a GUI. It is a simple command-line program that starts up and continues to run until it is closed.
- Upon starting up it will read the port it is supposed to operate on from a Properties file. The Server will then listen on that port for connections from Billboard Viewer and Billboard Control Panel applications.
- The Server's job is to keep track of users, billboards and schedules. This information is all stored in a MariaDB database.

- The Billboard Server will retrieve information about the database it is supposed to connect to from a Properties file named **db.props**. This includes the URL that JDBC is to connect on (jdbc.url), the database it will work from (jdbc.schema) and the username (jdbc.username) and password (jdbc.password) it will connect to the server with.
- For your own testing you will need to set up MariaDB and this properties file. For marking purposes your marker will provide a database and change the db.props file to provide the appropriate information.
- The database design is up to your team- however, you must have a minimum of three tables (one for users, one for the billboards and one for the scheduling) and all tables must be stored within the same database.
- If the Server is unable to connect to the database for whatever reason, an appropriate error message will be printed and the Server will terminate.
- On startup the Server will check the database to ensure that the tables it uses are present. If they are not, the Server will create them. Note that, when your project is marked, your marker will provide a freshly created database with no tables in it. Your Server will have to then create the necessary tables.
- On a fresh startup (where the Server has to create new tables) the Server will create a user (in your report you will give the username and password of this user so your marker can test your assignment) with full permissions, which can then be used to set up everything else and create new users.
- The Server will wait to receive connections from Billboard Viewers and Billboard Control Panels, over the port found in the networking Properties file.
- Connections will be short-lived "stateless" connections, similar to the HTTP protocol. That is, a Viewer or Control Panel will open up a connection to the Server and send data that describes what action that client wants the Server to take. The Server will then respond with information/acknowledgement and close the connection. The exact specifications of the protocol are left up to your team as a design decision.
- Billboard Viewers will only make one type of request- a request for the currently-showing billboard. The Server will then look at the schedule, determine which billboard is to be shown, then send the contents of that billboard over to the Viewer. Viewers will make these requests frequently (every fifteen seconds).
    - If there is no billboard scheduled at a particular time, the Server should send back something else for the Viewer to display in the meantime. Your team should decide on something appropriate.
- Billboard Control Panels will make many types of requests. More information about the functionality these requests are designed to serve is found under the 'Billboard Control Panel' section.
    - Login request: the Control Panel will send the Server a username and hashed password (see: User Authentication). The Server will either send back an error or a valid session token. *(Permissions required: none.)*
    - List billboards: the Control Panel will send the Server a valid session token and the Server will respond with a list of billboards (including the name and creator of each, as well as any other information your team feels is appropriate.) *(Permissions required: none.)*
    - Get billboard information: the Control Panel will send the Server a billboard's name and a valid session token. The Server will then send the billboard's contents. *(Permissions required: none.)*

- o Create/edit billboard: the Control Panel will send the Server a billboard's name, contents and a valid session token. The Server will then either create a new billboard (if the name is not already in use) or replace the contents of that billboard with the new one. *(Permissions required: to create a billboard, must have "Create Billboards" permission. To edit own billboard, as long as it is not currently scheduled, must have "Create Billboards" permission. To edit another user's billboard or edit a billboard that is currently scheduled, must have "Edit All Billboards" permission.)*
- o Delete billboard: the Control Panel will send the Server a billboard's name and a valid session token. The Server will then delete the billboard and any scheduled showings of it and send back an acknowledgement. *(Permissions required: if deleting own billboard and that billboard is not currently scheduled, must have "Create Billboards" permission. To delete any other billboards, including those currently scheduled, must have "Edit All Billboards" permission.)*
- o View schedule: the Control Panel will send the Server a valid session token and the Server will respond with a list of billboards that have been scheduled, including each billboard's name, creator, time scheduled and how long it has been scheduled to run for. *(Permissions required: "Schedule Billboards".)*
- o Schedule billboard: the Control Panel will send the Server a billboard name, time, duration and a valid session token (as well as any other information necessary, e.g. for handling recurrence.) The Server will then schedule the billboard to display at that time and send back an acknowledgement. *(Permissions required: "Schedule Billboards".)*
  - ▪ Note that, if another billboard is already scheduled at the same time or at an overlapping time, the newly-scheduled billboard will take precedence. e.g. if billboard A is scheduled from 6pm to 7pm and billboard B is then scheduled from 6:30pm to 7pm, at 6pm billboard A will display, then at 6:30pm it will change to billboard B.
- o Remove billboard from schedule: the Control Panel will send the Server a billboard name, time and valid session token. The Server will then remove this showing of that billboard from the schedule and send back an acknowledgement. *(Permissions required: "Schedule Billboards".)*
- o List users: the Control Panel will send the Server a valid session token and the Server will respond with a list of usernames (and any other information your team feels is appropriate.) *(Permissions required: "Edit Users".)*
- o Create user: the Control Panel will send the Server a username, a list of permissions, a hashed password and a valid session token. The Server will then create that user and respond with an acknowledgement. *(Permissions required: "Edit Users".)*
- o Get user permissions: the Control Panel will send the Server a username and valid session token. The Server will respond with a list of that user's permissions. *(Permissions required: if a user is requesting their own details, none. To get details for other users, "Edit Users" permission is required.)*
- o Set user permissions: the Control Panel will send the Server a username, a list of permissions and a valid session token. The Server will then change that user's permissions to match the new list and send back an acknowledgement. *(Permissions required: "Edit Users". Note that no user has the ability to remove their own "Edit Users" permission.)*
- o Set user password: the Control Panel will send the Server a username, hashed password. The Server will then change that user's password and send back an

acknowledgement. *(Permissions required: if setting own password, none. To set the password of another user, "Edit Users".)*

- o Delete user: the Control Panel will send the Server a username and a valid session token. The Server will then delete that user and send back an acknowledgement. Note that this may result in existing billboards no longer having an owner registered on the system. Your team should decide on something appropriate to happen under that circumstance. *(Permissions required: "Edit Users". Note that no user has the ability to remove themselves.)*
- o Log out: the Control Panel will send the Server a valid session token. The Server will then expire that session token and send back an acknowledgement. *(Permissions required: none.)*

- With all of the above transactions, the Server should send back appropriate error notifications to the Control Panel if the task should not be completed for some reason, such as:
  - o The session token is invalid or has expired. The user will need to log in again.
  - o The user does not have the right permissions to perform the action.
  - o The action is invalid for some other reason (attempting to get details about a non-existent user or billboard, for example.)
- The Control Panel can then notify the user of the error in an appropriate fashion.

User Authentication

- The verification of the username and password will take place over the network:
  - o The Control Panel knows what username and password the user has entered.
  - o The Server knows which users are valid.
- The password **must not be sent over the network**. Instead, it must be hashed (a form of one-way encryption) and sent over the network, to be verified by the server.
- The passwords must also be **salted**. This means that, when a password is stored on the server, a random string (the salt) must be generated and combined with the password after hashing. The resulting string must then be hashed again before being stored in the database alongside the salt.
- When the Control Panel user enters a username and password, the password will be hashed and sent across to the Server along with the username. The Server will then look up the salt associated with that user, add the salt to the hashed password, then hash the result again. If the final result is equal to the password stored in the database, the user is authenticated.
- If the user is successfully authenticated, the Server will generate a random session token string and send it back to the Control Panel.
- The Control Panel will send this session token along with each subsequent message, which the Server will use to verify that the user has been logged in correctly.
- If more than 24 hours have passed since the last successful use of a session token, that token will expire, and the user will need to log in again.
- The Control Panel will also feature a logout button, which a user can select to log out. This will cause the session token to expire immediately and the user will need to log in again before they are able to continue accessing the system.

Database Properties

- The Billboard Server will store persistent records (not including session tokens- these should be kept in memory) in a MariaDB database, the details of which will be retrieved from a file named **db.props**.

- The **db.props** file that the Server reads will have the following format (this is the same format used in the Database practical exercises. You are welcome to borrow the code from that practical and use it in your assignment):

```
jdbc.url=jdbc:mysql://localhost:3306
jdbc.schema=cab302
jdbc.username=root
jdbc.password=
```

- Although the corporation may end up using another database in the future, for now we have been instructed to use MariaDB. Your project should therefore include the MariaDB Connector/J JDBC driver as a dependency.

Billboard XML Format

- The contents of a billboard are described in an XML file. Your software will be *required* to use this file format for the purposes of allowing the user to export a billboard to XML and import a billboard from XML.
- Although it is not required, you *may* use this same format for storing billboards in your database and for transferring billboards over the network (between the Server and Control Panel and between the Server and Viewer.
- Here is an example billboard XML file:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<billboard background="#0000FF">
    <message colour="#FFFF00">Welcome to the ____  Corporation's Annual
Fundraiser!</message>
    <picture url="https://example.com/fundraiser_image.jpg" />
    <information colour="#00FFFF">Be sure to check out https://example.com/ for
more information.</information>
</billboard>
```

- Billboards are all UTF-8-encoded XML 1.0 files.
- Each billboard file has a single root node with the tag name **billboard**.
  - If the billboard node has a 'background' attribute, this will be interpreted as the colour to display for the entire billboard's background (that is, all parts of the screen that are not text or an image.) If the 'background' attribute is not supplied, use an appropriate default.
- Within the **billboard** node will be between 1 and 3 child nodes out of the following:
  - **message**: Used to show the primary text of the billboard. This text will be displayed in a large font size so it will be clearly visible- however, the text must all fit on one line, with no line breaks.
    - If the message node has a 'colour' attribute, this will be interpreted as the colour to display the message text using. If the 'colour' attribute is not supplied, use an appropriate default.
  - **picture**: Used to show a picture, which will be scaled to an appropriate size and displayed on the billboard.
    - The picture node will have one of two attributes: url and data. A picture node with neither of these attributes is invalid, as is a picture node with both.
    - If the url attribute is present, it will contain a URL pointing to the image, which may be of BMP, JPEG or PNG format:

```
<picture url="https://example.com/fundraiser_image.jpg" />
```

- If the data attribute is present, it will contain a Base64-encoded representation of the bytes making up an image- again, in BMP, JPEG or PNG format:

```
<picture
data="iVBORw0KGgoAAAANSUhEUgAAAAgAAAAICAIAAABLbSncAAAALHRFWHRDcmVhdGlvbiBUa
W1lAE1vbiAxNiBNYXIgMjAyMCAxMDowNTo0NyArMTAwMNQXthkAAAAHdElNRQfkAxAABh+N6nQI
AAAACXBIWXMAAAsSAAALEgHS3X78AAAABGdBTUEAALGPC/xhBQAAADVJREFUeNp1jkEKADAIwxr
//+duIIhumJMUNUWSbU2AyPROFeVqaIH/T7JeRBd0DY+8SrLVPbTmFQ1iRvw3AAAAAElFTkSuQm
CC" />
```

- When creating/editing billboards in the Billboard Control Panel, both options will be available:
  - The user can specify a URL to an image. If this is done, a URL-format picture tag will be inserted into the billboard.
  - The user can choose to browse their file system and find a BMP, JPEG or PNG format image. If this is done, the image will be converted to Base64 and stored in a data-format picture tag.
- o **information**: Used to show larger amounts of text information. This text can be broken across multiple lines for display purposes. This text should be shown at a smaller font size than the message tag.
  - If the information node has a 'colour' attribute, this will be interpreted as the colour to display the information text using. If the 'colour' attribute is not supplied, use an appropriate default.
- The placement and scaling of text and images will be dependent on the number of these tags that are used in a billboard, as any combination of the three tags may be present.
  - o If only **message** is present, the message should be displayed almost as large as possible, within the constraints that the text cannot be broken across multiple lines and it must all fit on the screen.
  - o If only **picture** is present, the image should be scaled up to half the width and height of the screen and displayed in the centre.
    - Note that this scaling up should not distort the aspect ratio of the image. If the screen is 1000 pixels wide and 750 pixels high, a 100x100 image should be displayed at 375x375. On the other hand, a 100x50 image should be displayed at 500x250. In each case the image is scaled, preserving the aspect ratio, to the largest size that can fit in a 500x375 (50% of the screen's width and height) rectangle.
  - o If only **information** is present, the text should be displayed in the centre, with word wrapping and font size chosen so that the text fills up no more than 75% of the screen's width and 50% of the screen's height.
  - o If only **message** and **picture** are present, the picture should be the same size as before, but instead of being drawn in the centre of the screen, it should be drawn in the middle of the bottom 2/3 of the screen. The message should then be sized to fit in the remaining space between the top of the image and the top of the screen and placed in the centre of that gap.
  - o If only **message** and **information** are present, the message text should be sized to fit in the top half of the screen and the information text sized to fit in the bottom half of the screen.

o   If only **`picture`** and **`information`** are present, the picture should be the same size as before, but instead of being drawn in the centre of the screen, it should be drawn in the middle of the top 2/3 of the screen. The information text should then be sized to fit in the remaining space between the bottom of the image and the bottom of the screen and placed in the centre of that gap (within the constraint that the information text should not fill up more than 75% of the screen's width.)

o   If **`message`**, **`picture`** and **`information`** are all present, all three should be drawn. The picture should be drawn in the centre, but this time at 1/3 of screen width and screen height (once again, scaled to preserve aspect ratio). The message should be sized and centred to fit in the gap between the top of the picture and the top of the screen. The information should be sized and centred to fit in the gap between the bottom of the picture and the bottom of the screen.

Unit Testing

- While this is an application involving GUI components, some of which may be problematic to unit test, you are expected to do unit testing where possible to help produce a high quality codebase.
- Unit testing is done with JUnit 5. Your test classes will be named using the convention of Test_____, where _____ is the same of the class you are testing.
- The unit tests must all be able to be run without requiring any external resources. This includes the database, all properties files and other applications (in other words, unit-testing classes in the Billboard Control Panel does not require the Billboard Server to be running. Unit-testing the Server will not require MariaDB to be running etc.)
  - o   This will require you to make use of mock objects to replace external resources. Your project should be architected to support this.
- Your marker will mark your unit testing, both by manually inspecting your unit tests and by running your unit tests and analysing code coverage.

Test-Driven Development

- Your team was chosen for this project partly because of your team's immense amount of experience with test-driven development. The client corporation was very interested in this, and to mollify them, you must **implement at least one moderately sized class using test-driven development as an approach.**
- In your report you must identify which class was implemented using test-driven development and describe the approach you used. Your marker/the client corporation will use your Git logs to corroborate this, so ensure that your commit history clearly shows this (e.g. by having the test class written first with the actual class only a stub, then later the class' methods are implemented.)

Report

- Your team must present a report along with the software. The report is **essential** because it will be used to guide the marking. If there is no report presented, very little of your assignment will be able to be marked.
- Your report will need to contain at least the following:
  - o   A statement of completeness, containing a high-level overview of what features were implemented in your submitted project.

- o A statement of contributions, containing the names and student numbers of each member of your team, as well as a statement describing what each team member contributed to the final result.
- o A high-level design description, explaining what each class is responsible for and how it interacts with the other classes in your system.
- o A section explaining how you used test-driven development and which class(es) was/were created using it.
- o Necessary setup/installation instructions, including how to create the network Properties file, the default user's username and password and anything else you feel the client corporation/your marker will need to properly assess your software.
- o A walkthrough of your system, describing how to set up and test each part of it. **This section is crucial** as the marker will go through your walkthrough and give you marks for each piece of functionality they see. **You will not get marks for anything you leave out.**
  - ▪ You may find it useful to make use of screenshots in this walkthrough, however, this is not mandatory.

Source Control

- Your team is required to use a Git repository to manage your project's source code.
- Each team member must make commits under their own name (it does not need to exactly match your real name, but the marker must be able to identify the student responsible for each Git commit.)
- You will be required to submit your entire Git repository (each Git repository contains a hidden '.git' folder that is necessary as it contains the full commit history of the project. You may need to make some configuration changes to your operating system to be able to see this. On Windows, "Send To → Compressed (zipped) folder" appears to work well.)

Source Code Documentation

- You are required to document your source code with both regular comments (// and /* */) and JavaDoc-style comments (/** */).
- Regular comments go in the body of your methods and describe the intention of the code they precede.
- JavaDoc-style comments precede your classes and methods and thoroughly document the semantics of that class/method, the arguments they take, any return values and any exceptions thrown.