Written report

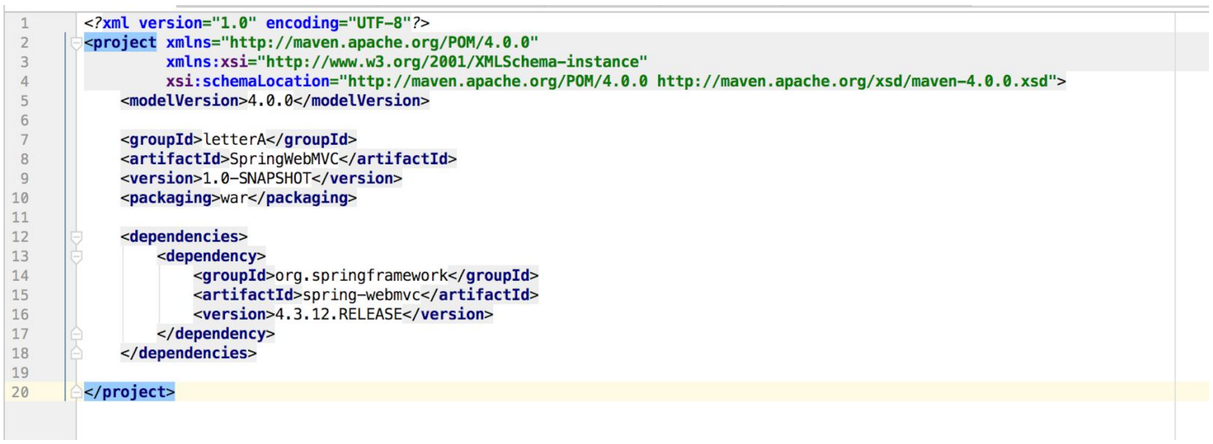
The length of report will partly depend on your team size *n*.

Part 1. Review of the technology [6 marks]
The structure of this part is fairly flexible, but it should answer the following questions:

1. **What features/benefits the technology provides? Pick *n* that your team thinks are most relevant and elaborate on each**

    1. **Spring Security** is a framework that focuses on providing both authentication and authorization to Java applications. It has these features: comprehensive and extensible support for both Authentication and Authorization; protection against attacks like session fixation, clickjacking, cross- site request forgery; optional integration with Spring Web MVC.
    2. **Feature update** can be used by any Java application. Spring comes with some of the existing technologies like ORM framework, logging framework, J2EE and JDK Timers etc. Hence we don't need to integrate explicitly those technologies.In this case, Spring is adaptable to any Java application and becomes popular.
    3. **Open source**, Java spring enrolls the most famous open source foundation "Apache". Also Java itself is a big community and Spring can be widely used based on Java application.
    4. **Spring application can be used for the development of different kind of applications**, like standalone applications, standalone GUI applications, Web applications and applets as well.
    5. **Spring enables the developers to develop the enterprise applications using *POJOs*** (Plain Old Java Object). The benefit of developing the applications using POJO is, that we do not need to have an enterprise container such as an application server but we have the option of using a robust servlet container.

2. **What features are you demonstrating in your demo website? (provide links and/or screenshots to guide the reader)**
    Dependencies and package management with large java application



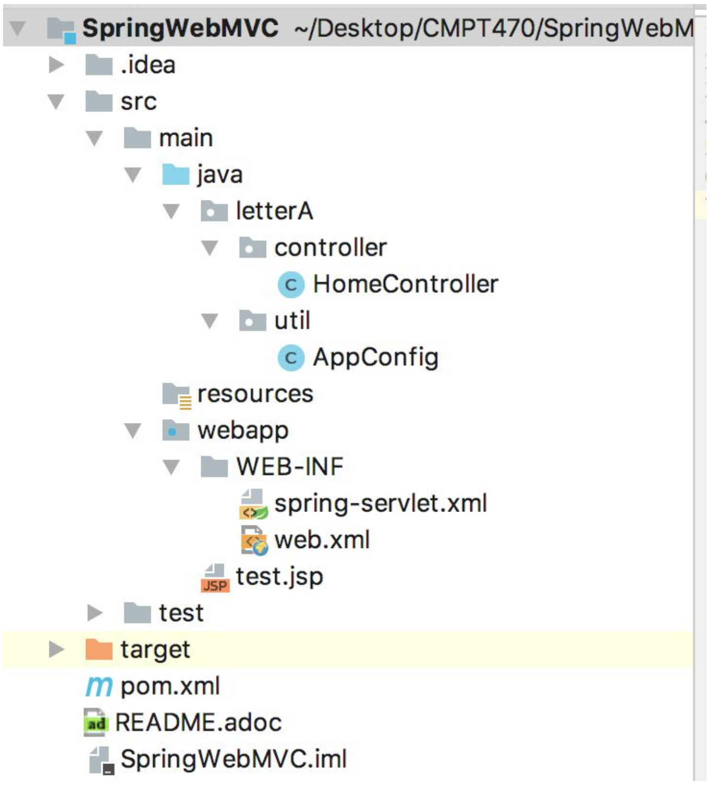
```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>letterA</groupId>
    <artifactId>SpringWebMVC</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>war</packaging>

    <dependencies>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>4.3.12.RELEASE</version>
        </dependency>
    </dependencies>

</project>
```

MVC:



```
SpringWebMVC ~/Desktop/CMPT470/SpringWebM
  .idea
  src
    main
      java
        letterA
          controller
            HomeController
          util
            AppConfig
      resources
      webapp
        WEB-INF
          spring-servlet.xml
          web.xml
        test.jsp
    test
  target
  pom.xml
  README.adoc
  SpringWebMVC.iml
```

Open source related:
SDK, Maven, Tomcat

Part 2. Comparison with existing technologies [6 marks]
Compare the technology with at least *n*−1 comparable technology alternatives suitable for similar applications. This comparison should be roughly 200 x *n* words in length (must not exceed 200 x 2*n* words).

JAVA Spring vs Node.js

A key element of Spring is infrastructural support at the application level: Spring focuses on the "plumbing" of enterprise applications so that teams can focus on application-level business logic, without unnecessary ties to specific deployment environments.

Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Modern Java languages can start by being synchronous and spawn asynchronous processes at the same time (depending on the method, class and object being called) and even make use of multiple cores so they can literally spawn 'new waiters' as needed if they get a rush of customers.

Node likes to brag about being asynchronous (like a waiter in a restaurant being able to serve multiple customers at a time in a restaurant and wait on others in its downtime) while claiming Java can only be synchronous. This is hardly true.

There is a critical choice that can greatly impact your platform decision. What is your chosen storage mechanism. If it's a relational store the clear choice is Spring. If it's Mongo or some other document store, Node is the stronger candidate. Java’s ability to do relational store is unmatched compared to Node.

JAVA Spring vs Jakarta Struts(or 2)

Struts Framework is the product of the Apache software foundation which basically used for developing web applications in Java. Struts framework resides in the middle tier of JSE, JEE and JME; which is Java Enterprise Edition. As this framework is provided by apache; it is also known as Apache Struts or Jakarta Struts. It was originally known as WebWork framework.

Struts framework follows the basic MVC pattern. The controller has two parts: System Core Controller and Business Logic Controller. System core controller is corresponding to the ActionServlet part in MVC, provides by Struts framework, inherited HttpServlet category, so it can be configured as a marked Servlet. Business Logic Controller is responsible for handling user requests, but it cannot process by itself, call the Model to complete the processing instead. It corresponding to the Action in the MVC.

Most people use EJBs or even Spring JavaBeans as their model; however, the view is important in Struts(2) because it has its own view language. This is an advantage over SpringMVC, so that we can build a Java Framework from beginning to end by Struts(2).

To compared with Struts, Spring provides a very clean division between controllers, JavaBean models and views, and it is very flexible. Spring makes application loosely coupled by using Dependency Injection, the core of the Spring framework is the IoC container. Different from Struts, Sprince could force the Action and Form objects to concrete inheritance. Spring MVC is entirely based on interfaces and it like

WebWork provides interceptors as well as controllers, making it easy to factor out behaviour common to solve different types of requests.

We can conclude that the main difference between struts and Spring MVC is about the difference about the AOP and OOP. OOP can do everything AOP does but with different approaches.

JAVA Spring vs Hibernate

Hibernate is an open source object relational mapping framework, encapsulates JDBC with a very lightweight object. It allows Java programmers to use the databases using object-programming method as they want.

In Hibernate, it has 5 core interfaces: Session, SessionFactory, Configuration, Transaction and Query & Criteria.

1. Session interface is responsible for executing the CRUD operations of the persisted but not thread-safe object.
2. SessionFactory interface is responsible for initializing Hibernate and creating session objects. It acts as a proxy for the source of data storage.
3. Configuration interface is responsible for configuring and starting Hibernate. While Hibernate starting up, an instance of the Configuration class locates the mapping document first, then reads the configurations, following by creating SessionFactory Object
4. Transaction interface is an optional and it is responsible for transaction-related operations.
5. Query and Criteria interface is responsible for implementing different types of database queries (i.e HQL language, SQL statement)

To compare between Hibernate and Spring, we can conclude as following:

First, Hibernate is an ORM tool for data persistence. Hibernate can be used in EJB J2EE architecture to replace CMP for data persistent.).

Next, Hibernate provides lots of feasibility while dealing with the database. (Hibernate can specify a SessionFactory for each of database while operating multiple databases). Hibernate could provide with ORM where in database transaction is the form of objects mapped to corresponding table using configuration xml files.

To be summary in one sentence between these two technologies, Spring is a framework with IoC and AOP. Hibernate could use to connect with database with ORM.

JAVA Spring (MVC) vs Spring Boot

In general, Spring MVC is a complete HTTP oriented MVC framework that managed by the Spring framework and based in Servlets. It is similar or same as JSP in the

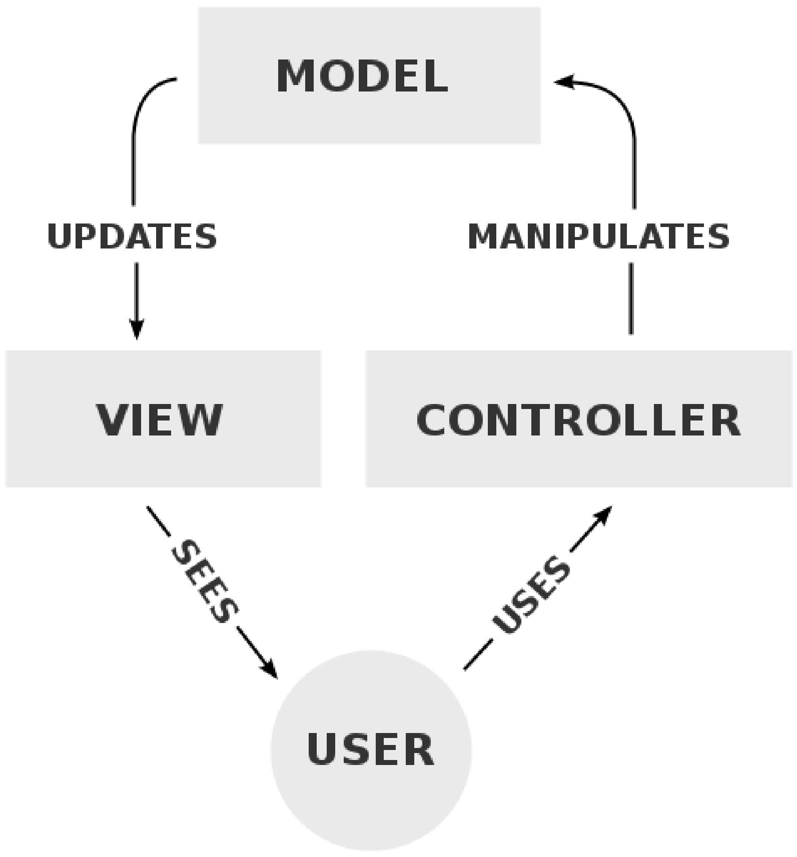
JavaEE stack. “Controller” is the most used elements in it where we can implement method to access using different HTTP requests.

Spring boot, is a utility for setting up applications quickly, it offers an out of the box configuration in order to build Spring powered applications. Spring integrations cover a large area for different modules in its umbrella, (as spring-core, spring-data, spring-web), which includes Spring MVC as well. With Spring boot, we can tell Spring how many of them we want to use and we will get a fast setup.

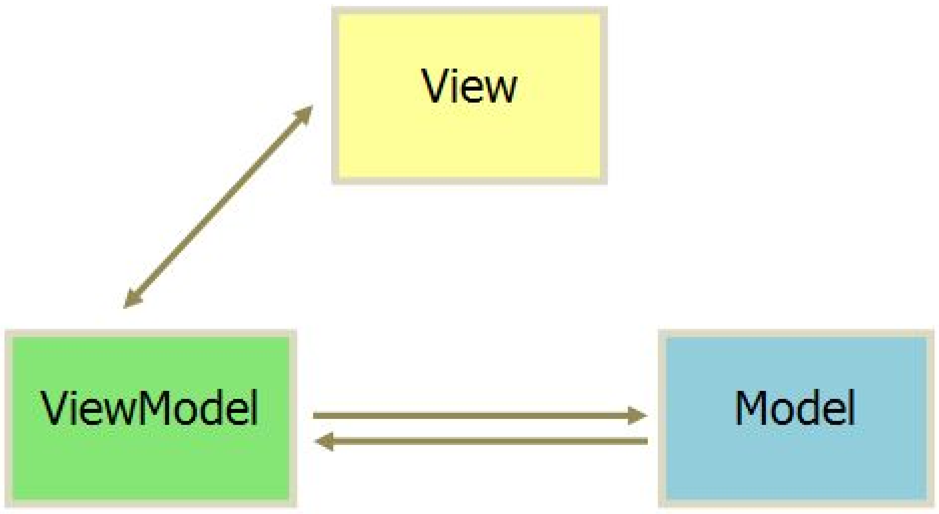
To be conclude the these two technologies, Spring MVC is a framework to be used in web applications and Spring boot is a Spring-based production-ready project initializer. They played different roles in Spring as they are relatives in Spring family.

Part 3. What your team (re)learned most useful for your final course project? [4 marks]
List 3 x *n* items that your team has learned most useful that your team will try to adopt for your final course project.
The items can be general concepts, design/development principles, concrete examples demonstrating the terms/theories presented in class. For each item, elaborate with 3-5 sentences.

1. MVC
   Model–view–controller (MVC) is an architectural pattern commonly used for developing user interfaces that divides an application into three interconnected parts. This is done to separate internal representations of information from the ways information is presented to and accepted from the user. The MVC design pattern decouples these major components allowing for efficient code reuse and parallel development.



2. MVVM
   MVVM facilitates a separation of development of the graphical user interface – be it via a markup language or GUI code – from development of the business logic or back-end logic (the data model). The view model of MVVM is a value converter,[1] meaning the view model is responsible for exposing (converting) the data objects from the model in such a way that objects are easily managed and presented. In this respect, the view model is more model than view, and handles most if not all of the view's display logic.[1] The view model may implement a mediator pattern, organizing access to the back-end logic around the set of use cases supported by the view.



3. Bootstrap
   Bootstrap provides an easy-to-use framework of ready-made styles, layout tools, and interactive components, allowing developers to create websites and applications that are visually appealing, functionally rich, and accessible out of the box.
4. Two-way data binding
   Data binding in AngularJS is the synchronization between the model and the view. When data in the model changes, the view reflects the change, and when data in the view changes, the model is updated as well. This happens immediately and automatically, which makes sure that the model and the view is updated at all times.
5. jQuery
   jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

6. HQL and SQL

SQL is a standard language for storing, manipulating and retrieving data in database. HQL is a query language which is similar to SQL.To compare with SQL, HQL is fully object-oriented and understands notions like inheritance, polymorphism and association.

7. AOP (Aspect Oriented programing)

A programming paradigm learned from Java Spring, which increases modularity by separating problem into distinct parts. It adds additional abilities to functions without adding extra code.

8. MVW

AngularJS using MVW architecture allows user to create “whatever” works , where “whatever” stands for whatever works for you. It’s close to MVVM, object could be considered the ViewModel that is being decorated by a function that we call a Controller.

9.Java servlet

a Java program that extends the capabilities of a server. Although servlets can respond to any types of requests, they most commonly implement applications hosted on Web servers.Such Web servlets are the Java counterpart to other dynamic Web content technologies such as PHP and ASP.NET. And servet embed HTML inside Java code instead of Java code inside HTML.

10.View Template

Viwe Templat is a convenient way to generating dynamic HTML page. As a part of web framework architecture, template system renders data to appropriate place, this allows for reuse of static elements of a web page and dynamic element to be defined based on web requests.

11. Responsive design

Responsive web design is an approach to web design which makes web pages render well on a variety of devices and window or screen sizes. Recent work also considers the viewer proximity as part of the viewing context as an extension for RWD.

12. JSX

Instead of artificially separating technologies by putting markup and logic in separate files, React separates concerns with loosely coupled units called “components” that contain both. We will come back to components in a further section, but if you’re not yet comfortable putting markup in JS, this talk might convince you otherwise.

13. The Virtual DOM

A virtual representation of the DOM. Any new view changes are first performed on the virtual DOM, which lives in memory and not on your screen. An efficient algorithm then determines the changes made to the virtual DOM to identify the changes that need to be made to the real DOM. It then determines the most effective way to make these changes and then applies only those changes to the real DOM.

This guarantees a minimum update time to the real DOM, providing higher performance and a cleaner user experience all around.

14. Component-Based

React build encapsulated components that manage their own state, then compose them to make complex UIs.

15. Declarative

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes. Declarative views make your code more predictable and easier to debug.

Appendix [4 marks] (this section is required)

A summary list of repositories (of existing web apps) your team has cloned (URL) and a summary of the extensions your team made. Your Git commits should support this summary list. This section is meant to bring special attention to Jay and Lisa on things your team would like to highlight.

https://csil-git1.cs.surrey.sfu.ca/liyujiel/letterA470Demo

We used various open source softwares to support this demo for different purposes, include package management(SDK), Java servlet(Tomcat), Java automation build tool(Maven).