Chapter 14
SelecWeb: A Software Tool for Automatic Selection of Web Frameworks

Thaha Muhammed, Rashid Mehmood, Ehab Abozinadah, and Sanaa Sharaf

14.1 Introduction

The software applications revolution has helped the development of many new distributed and collaborative urban systems [1–16], paving the way for integrated systems, and hence smart cities and societies, see, e.g., [17] for background on smart cities and societies.

Web applications and services are fundamental to designing smart infrastructure and cities. Web frameworks have become an integral part in the development of web applications and services. A software framework is a scaffold structure inside which other applications can be developed. A framework comprises libraries, services, scaffold programs, scaffold codes, interfaces, APIs (application programming interfaces), and other components required for application development. A framework provides the basic building blocks required for the development of application.

A software framework that has been developed particularly for assisting web application development is called a web application framework. It comprises necessary components and services required for the construction of feature rich applications by automating the common web development functions. Most of the web application development frameworks implement the MVC (model–view–

T. Muhammed (✉) · S. Sharaf
Department of Computer Science, FCIT, King Abdulaziz University, Jeddah, Saudi Arabia
e-mail: m.thaha.h@ieee.org; ssharaf@kau.edu.sa

R. Mehmood
High Performance Computing Center, King Abdulaziz University, Jeddah, Saudi Arabia
e-mail: rmehmood@kau.edu.sa

E. Abozinadah
Department of Information Systems, FCIT, King Abdulaziz University, Jeddah, Saudi Arabia
e-mail: eabozinadah@kau.edu.sa

R. Mehmood et al. (eds.), *Smart Infrastructure and Applications*,
EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-030-13705-2_14

controller) design pattern as shown in Fig. 14.1. It also incorporates various services like versioning using git, svn, or other version management systems and searching. This helps the application in leveraging the framework services for the production of quality applications. Web frameworks also provide user interface elements and powerful ORM (object-relational management).

Web application frameworks promote code reuse and reduce the resource requirements such as time and effort to build and maintain applications. In recent years, a plethora of frameworks have been developed with various features. There is no single all-feature-encompassing framework. Each framework has its own advantages and disadvantages. Suitability of various web frameworks to various application domains varies. Programmers may choose from a variety of web frameworks, and different languages that support them, each with its own strengths and weaknesses. Organizations work in different application domains and have diverse priorities and constraints with regard to the development of applications and services.

In this paper, we propose an automatic tool for selecting a web framework based on a set of criteria and developer preferences. The set of selection criteria is developed by us and is a contribution of this paper. The tool is called SelecWeb. It currently uses analytic hierarchy process (AHP) for comparison, analysis, and decision-making. We provide a detailed description and analysis of the tool including a case study for web framework selection.

The rest of the paper is organized as follows: Sect. 14.2 gives a review of the literature. Section 14.3 introduces the web frameworks that we have been used in this paper. These are Ruby on Rails, Spring, Django, and CodeIgniter. Section 14.4 discusses the selection criteria including the developer and user criteria. Section 14.5 explains the evaluation process and discusses evaluation of each of the selected frameworks. Section 14.6 summarizes the weaknesses and strengths of each

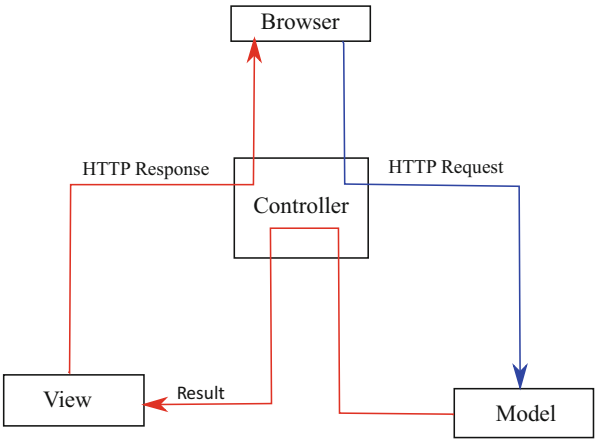

Fig. 14.1 The MVC architecture for web frameworks

framework. Section 14.7 provides a case study that uses AHP for selecting the best framework satisfying requirements of a given application. Section 14.8 concludes the paper along with some design ideas on future extension of the tool using machine learning.

14.2 Literature Survey

Many researchers have evaluated single frameworks in isolation. Numerous works evaluate primary technologies for web applications such as HTML5 [18]. Analysis of web framework application development often evaluates a single web framework, but do not compare it to competing ones. Bachle and Ritscher [19] analyzed and benchmarked Rails. Arthur and Azadegan [20] assessed Spring, a Java-based framework. The researchers evaluate the framework and do not compare it to other competing frameworks. These works mainly list out the features and capabilities of the framework. Matt [21] compares various Java-based frameworks such as Spring, Wicket, Grails, Play, and JRuby. A comparison of Eucalyptus, Apache Hadoop, and the Django–Python stack can also be found [22]. These comparisons deal with frameworks based on the same language. Smutny [23] briefly compares selected web-based mobile frameworks. However, he does not propose a set of criteria for doing so. Henning et al. make a comparison of four mobile web frameworks [24], where they compare HTML5, Sencha Touch, Google Web Toolkit, M-Project, jQtouch, and jQuery. Here, the comparison is based on a set of criteria developed by the researchers. We can reach a conclusion that most of the comparisons lack the backing of scientific criteria.

14.3 Web Frameworks

This section examines the web application framework and introduces four frameworks that will be analyzed in Sect. 14.5.

14.3.1 General

A web framework is a collection of packages and modules that helps web developers to write web applications, web services, and dynamic websites without worrying about low-level details [25]. Frameworks ease the overhead associated with common web development activities. Many frameworks provide libraries for database access, provide support for a number of activities such as interpreting requests, templating frameworks, producing responses, manage sessions, and they promote code reuse often [26].

Many of the major frameworks such as Rails, Django, and Spring are server-side technologies but in the recent years due to advancements in client-side technology such as node.js and CoffeeScript, browsers can be used as a full blown framework stack.

World Wide Web in its infancy used static pages hand coded in HTML which were hosted on web servers. Any alteration to the website required explicit changes from the developers [25]. In earlier days, Common Gateway Interface [25] was used to serve web pages to the web browser. In CGI, we had to program each and every detail of the connection. Each request that arrives at CGI creates a new thread. As the number of requests increases, the number of threads also increases, which might crash the server. New languages for web development like PHP emerged around same time.

Most of these web languages used a spaghetti styled coding where everything starting from HTML views, database access, and other low-level details such as protocols, thread management, and socket management had to be hand coded [26]. These required various libraries that usually did not come with the language and had to be compiled by the developer.

Later on, came frameworks that constitute necessary components and services required for the construction of feature affluent and erudite systems. For instance, Ruby on Rails, Django, Symfony, Zend, CakePHP, CodeIgnitor, Spring, Grails, etc., are some web frameworks.

14.3.2 Ruby on Rails

Rails is a web framework written in Ruby by David Heinemeier Hansson [27]. He derived it from Basecamp, formerly Signal37, a project management tool. Rails increases the productivity of the developer by reducing the line of codes to achieve the end result. It accomplishes more in the least number of lines as compared to other languages such as PHP. Rails is based mainly on two principals: convention over configuration and DRY (don't repeat yourselves). It has an MVC (model–view–controller) architecture. Rails presumes that there is a perfect way to code, and sets these as conventions to be followed while coding in Rails. This results in higher productivity as configuring each and every minute configuration of the application is not required. The second principle called DRY (don't repeat yourselves) states every unique function should only have a single piece of code that accomplishes the task. This results in a more maintainable and less buggy code. Rails is released under the MIT License. It was released in the year 2003. Twitter and Github are the two major websites created with Rails.

14.3.3 Django

Django is a python based free open-source web application framework that was created by Adrian Holovaty and Simon Willison. Currently, it is maintained by an organization called Django Software Foundation. Django has many similarities to Ruby on Rails. Django has an MVC architecture. A cardinal advantage of such a concept is that components are loosely coupled which implies that a database architect's work will not depend on the programmer's or the designer's work. All three can work independently. Amazon.com, craigslist.org, and washingtonpost.com are some of the major applications built with Django. Django is licensed under the BSD License.

14.3.4 Spring

Spring is a web framework which is also called the father of frameworks, due to the fact that it provides scaffold to other Java-based frameworks. Some of the major frameworks that receive support from Spring are Hibernate, EJB, Struts, JSF, etc. [28]. In 2003, Rod Johnson created Spring. Spring is a Java-based framework helpful in creating Java Enterprise applications. Spring combines various components. It is highly valuable when you might want to use different components or various combinations of components in different environments with various configurations. Spring is a framework based on a pattern called dependency injection, which is issued to build highly decoupled systems [28, 29]. Spring consists of an MVC framework, validation framework, and transactional control of databases. It segregates service layer, business layer, and web layer. But what it really does best is injection of objects. In dependency injection [30], the objects are designed such that they receive instances of objects externally from other sources, instead of creating them inside the actual code. This improves decoupling and simplifies testing.

14.3.5 CodeIgniter

CodeIgniter [28] is a web-based framework for building dynamic websites based on PHP [28]. It was created by Rick Ellis in 2006, from ExpressionEngine, a CMS (content management system) owned by Ellislab in 2006. In 2014, the ownership of CodeIgniter was transferred to the British Columbia Institute of Technology, which inculcated it in their core syllabus. CodeIgniter is a lightweight, fast framework with a minimal footprint that helps in rapid application development. PHP's creator Rasmus Lerdorf, an outspoken critic of frameworks, praised CodeIgniter due to its speed and light weightlessness. It's loosely based on MVC architecture. It does not enforce the MVC pattern upon the developers. Controllers are necessary for the

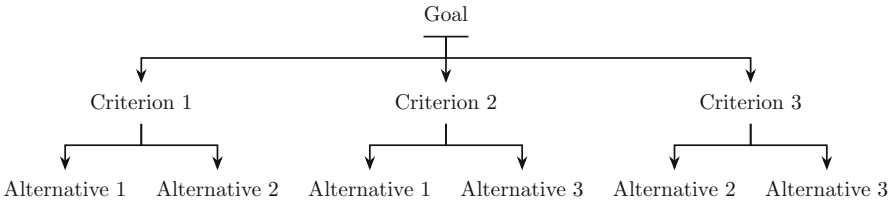

Fig. 14.2 Analytic hierarchy process

development, but models and views are optional, unlike other frameworks such as Rails and Django that enforce MVC strictly. It is a lean MVC framework, with enough capabilities to increase your productivity, at the same time providing third-party modules for extra functionality. The source code of CI is available online at Github [28]. Earlier, CI versions had an Apache-BSD open-source license. Later it was switched to the MIT License.

14.4 Selection Criteria

Web framework selection is an optimization problem [24] because we need to maximize the goals of the organization. We can measure complex requirement criteria by dividing them into sub-criteria and can use functions such as analytic hierarchy process [31] to evaluate final decision criteria as shown in Fig. 14.2. If the decision process is broken down into smaller manageable components, it results in an improved decision-making. Each of these criteria can be given a weight, according to the goals of an organization. Evaluating frameworks based on a single criterion is difficult and a vague measure. Hence, we need to develop a set of criteria, for the purpose of evaluation of frameworks. The overall goal of the decision criteria is to allow an organization to select an optimal framework based on their requirements. Hence, we divide the criteria into two, first from a developers' viewpoint, in terms of usability and second, from the users' viewpoint. A users' viewpoint and their experience with the application is of paramount importance. From developers' viewpoint, we consider the experience and decisions of the developer, such as licensing and cost. Criteria can be classified in two categories: qualitative and binary. Binary criteria are criteria that can be answered using either using yes or no. It examines whether a feature is available in the framework. Qualitative criteria deals with the quality of the framework. Qualitative criteria are extremely useful for decision making. Hence this article will deal with qualitative criteria.

14.4.1 Developer Criteria

From the viewpoint of the developer we consider the following criteria.

License and Cost Various companies have different policies regarding the license of the components and applications they use. Therefore, we need to consider the cost of licensing an application based on the web frameworks. Open-source licenses such as Apache [32] and MIT [33] can be considered as ideal cases, whereas complications can arise from the use of Copyleft licenses [34] such as GNU [35].

Learning Effort Effort and time are required to learn and comprehend a new web framework. This criterion examines the extent up to which the framework follows the general conventions, the intuitiveness of the framework, resemblance to other programming frameworks, and ease of learning. This also considers the effort and time required to master the framework.

Developing Effort The development effort is proportional to the cost of the development. Even though, requirement phase is independent of the framework used, it does influence the implementation. Development effort comprises of time required for the implementation of applications with the framework. Ease of reuse, good tool support, and an IDE with a graphical user interface are good indicators towards development effort.

Long-Term Viability Selection of a framework for development is a significant investment as all the applications developed by the organization will be tied to the framework. Due to rapidly changing technology the web frameworks require frequent updating. A framework with a robust team of developers commercially backed by organizations has greater potential to thrive. Popularity and frequent updates are two other indicators that imply long-term feasibility.

Documentation and Support Good documentation and support increase the speed of mastering the framework. A good quality documentation provides exceptional tutorials and references. Textbooks can be the starting point for popular frameworks. Forums and community provide extra assistance that helps the developer tremendously.

Adaptability Due to the evolution of the technology it may be necessary to modify the framework with extra functionality. This will be easier if the framework provides a module plugin mechanism. This criterion also evaluates the efficacy and availability.

Maintainability The code has to be maintainable. The source code has to be comprehensible and reusable. Modularity and decoupling of the components in framework increase the maintainability. These are the major indicators for maintainability.

14.4.2 User Criteria

From the viewpoint of the user, we can consider the following criteria.

Inherent Look and Feel The acceptance of a web application by a user mainly depends upon the look and feel of the application. Different frameworks provide various themes and feel. A framework should be able to provide a platform-specific theme. We see how the theme provided by the frameworks resembles the native look and feel unless the framework provides means to modify its user interface elements.

Load Time The web application has to be fast and load even on unstable and slow networks. The load time depends upon the complexity of the web framework. The web frameworks can use asynchronous JavaScript or cache the commonly used pages to increase the load speed. This criteria measures the load speed of a page for web frameworks.

Runtime Performance The total runtime performance of the application after loading is important. The dynamic page should respond quickly to user interaction. The user interface elements should react without any lag. The animations have to be smooth. These indicators create an impression on the user about the frameworks performance.

14.5 Evaluation

In this section we present the results of our evaluation. We provide the result of each framework in the following subsections, respectively. Table 14.1 gives the summary of the evaluation.

Table 14.1 The evaluation summary of the web frameworks against criterion

Criterion	Ruby on Rails	Django	Spring	CodeIgniter
License and cost	1	1	1	1
Long-term viability	1	1	1	3
Documentation and support	1	1	1	4
Learning success	1	1	3	3
Development effort	2	2	4	3
Modifiability	3	2	2	3
Maintainability	2	2	2	3
User interface	5	5	2	4
Look and feel	4	5	5	5
Load-time performance	3	3	5	1
Run-time performance	2	3	4	1

14.5.1 Evaluation Process

The evaluation process was divided into two phases. In the first phase data and information were collected about the framework to get an initial impression. Online documentations, manuals, and forums were utilized to achieve this. Criteria such as cost and license were assessed in this manner. In the second phase a prototype application, a To-do list, was developed using all the frameworks being tested. Based on this experience, a reviewer rated the frameworks on a scale from 1, excellent, to 6, inferior, for each criterion. In case where the framework selection needs to be done automatically, this information can be automatically acquired and inferred through, for example, crowd-sourcing process, or machine learning (training and prediction or decision-making). The summary of the evaluation can be seen in Table 14.1. Web framework selection is an optimization problem [24] because we need to maximize the goals of the organization. We can measure complex requirement criteria by dividing them into sub-criteria and can use functions such as analytic hierarchy process [31] to evaluate final decision criteria. If the decision process is broken down into smaller manageable components, it results in an improved decision-making. Each of these criteria can be given a weight, according to the goals of an organization. Evaluating frameworks based on a single criterion is difficult and a vague measure. Hence, we need to develop a set of criteria, for the purpose of evaluation of frameworks. The overall goal of the decision criteria is to allow an organization to select an optimal framework based on their requirements. Hence, we divide the criteria into two, first from a developers' viewpoint, in terms of usability and second, from the users' viewpoint. A users' viewpoint and their experience with the application is of paramount importance. From developers' viewpoint, we consider the experience and decisions of the developer, such as licensing and cost. Criteria can be classified in two categories: qualitative and binary. Binary criteria are criteria that can be answered using either using yes or no. It examines whether a feature is available in the framework. Qualitative criteria deals with the quality of the framework. Qualitative criteria are extremely useful for decision-making. Hence this article will deal with qualitative criteria.

14.5.2 Ruby on Rails

Ruby on Rails is released under MIT License, which backs both closed- and open-source projects. It is currently hosted on Github. There is no cost for extra support or other developmental tools. Hence, grade 1 for license and cost. Rails is still in active development and releases new versions and updates frequently. RoR is used by many notable firms such as Twitter, Shopify, and SoundCloud. Rails recently released version 4.2. These positive trends predict a good performance in near future in terms of long-term feasibility. Hence, rank 1.

The documentation provides detailed instructions and tutorial about all available features with detailed examples. Many popular textbooks, articles, and tutorials exist. Forums and stack overflow provides support for Rails. Hence, documentation and support is also evaluated to 1. RoR can easily be learned due to good quality of documentation. It is a highly intuitive language with a highly intuitive syntax, which is very near to the natural language. No extra concepts are required to learn Rails. Hence, we can rank it 1.

Static and dynamic applications can be developed quite easily. Most of the database side code is generated by the scaffold generator. MVC model makes it easier to code. There are many third-party IDEs, even though one is not required. One of the major IDEs is RubyMine from JetBrains. It provides various advanced functionality such as internationalization. It speeds up the development of web application rapidly. Hence, we can provide development effort a grade of very good (2).

Rails is highly modular in nature. It uses third-party plugins called as gems. Any functionality can be added to Rails using third-party gems. As these gems are third party, they are not well documented and hence causes difficulty in extending the software. Hence, extensibility is satisfactory (3). Rails is written using Ruby. An application can be separated to various components in Rails. Since it is based on conventions rather than configuration, Rails is easier to maintain. Therefore, maintainability is very good (2).

Rails by itself does not provide any user interface elements, other than the one supported by HTML, which is not visually appealing. You have to use third-party themes such as Bootstrap or Foundation to provide themes to various UI interfaces. Hence UI elements get a grade of not well fulfilled (5). Rails by default not provide a native look and feel. It will not change from platform-to-platform. Hence it gets a rating of satisfactory (4). The load time of the Rails application depends upon the number of jQuery scripts and CSS classes being loaded. If the pages are highly dynamic, then the performance slightly decreases. This can be alleviated by using minified jQuery. It runs animation fluently once loaded. The performance after loading is excellent. Hence, the load-time performance is satisfactory (3) and the runtime performance is very good (2).

14.5.3 Django

Django uses the 3-clause BSD License also known as modified or revised BSD License which supports both open-source and closed-source development (1). Django is maintained and developed by a non-profit organization called Django Software Foundation. Many major organizations such as Instagram, Mozilla, and Bitbucket use Django for their web application. One of the major goals of the organization is the long-term viability of Django framework. Django is actively developed and frequently released with new updates and bug fixes due to which the Django can be predicted to have a solid future (1).

The Django website provides a good documentation on all of its features along with a separate tutorials for beginners, intermediate, and advanced developers. It provides a good elucidation on all of the APIs provided Django. Thus, Django has an excellent documentation (1). Since Python is based on Python developers will have to learn Python to code, to code in Django. But Python is a simple language to master. But good documentation paves an easy path for learning (1). It has a good code scaffold that does a lot of code scaffolding that reduces the number of lines to be written. Database transactions are automatically handled by an ORM. But the visual elements including CSS and JavaScript has to be coded separately. As a result, the development effort is nearly minimal in Django (2). Django is modular in structure. You can add python packages to Django to extend the functionality. These packages are developed by third-party developers and is hosted at PyPi. Django has pretty straightforward code due to the simplicity of python. Therefore, Django has a very good extensibility (2). Due to modular design and comprehensible code, Django is maintainable (2).

Django doesn't come with any template engine or user interface elements. We need to use third part party templates or custom CSS to provide it exceptional looks. Hence, the look and feel of Django can be rated as poor (5). Moreover, it does not support native look and feel, which would require higher customization using CSS (5). Django is a big framework with a size of 7.5 MB. Applications built with Django tend to be large in size. It provides an extra admin panel which provides a complete control of your application. Django loaded pretty fast (2) and had a very good runtime (2).

14.5.4 CodeIgniter

CodeIgniter is licensed under MIT License which supports both open- source and closed-source software. Earlier it was licensed under Apache/BSD-style open-source license. Due to GPL, incompatibility of the license was shifted to MIT License (1). It is currently maintained and developed by students of the British Columbia University. Hence licensing of CodeIgniter is excellent, but since it is developed in an academic environment without any commercial support its future is not very bright (3).

The documentation of CodeIgniter provides basic coverage of all features and a small tutorial. Initial learning curve is small but later on it becomes quite difficult to master complex development scenarios. The online help is not advanced. There are third-party tutorials and books for mastering CodeIgniter. Hence for long-term feasibility it is satisfactory (4) and documentation and support is average (3). CodeIgniter is based on PHP. It is based on MVC framework. It is easy for the beginners to get started. But mastering requires quite an effort (3). Various integrated development environments are available for the development of PHP such as Zend, NetBeans, and Eclipse. Development effort is low for a simple project, but it becomes harder for complex projects, but it doesn't require any configuration. It is a zero configuration framework. Hence, the net development effort is average (3).

When the project becomes large and complex it becomes difficult to maintain and extend the application mainly due to the fact that PHP consists of spaghetti code. But simple plugins can be added to most of the simple functionalities. Hence, the extensibility is average (3). Maintainability of large and complex applications is going to be a problem. Hence, the maintainability is average (3).

CodeIgniter doesn't provide any user interface elements. It uses the normal elements provided by HTML. Any further modification requires third-party themes that are available as modules. We can use CSS to liven up the application but it requires greater development effort (4). Moreover, it doesn't provide native look and feel. Its look and feel is independent of the platform (5). CodeIgniter is the smallest framework of all the frameworks considered in this test. It just has a size of 2.5 MB. It has a minimal footprint and hence it has fastest loading time (1). Runtime performance is excellent for CodeIgniter (1).

14.5.5 Spring

Spring is licensed under Apache 2.0. Apache 2.0 is a copyright license that is compatible with both open source and closed source, so the license criterion has been fulfilled (1). The Spring software is a huge framework made for enterprises and is usually considered as an alternative for Java Bean. It is currently maintained by Pivotal Software which is a large enterprise that creates software such as VMware. Major release rolls out every year with new features and fixes. Since it is supported by a large corporation, and the code is open source, it has a very strong solid future with high potential (1).

Spring has very detailed and long documentation of all its features. Since it is an enterprise based MVC framework, it has a lot of documentation that needs to be learned. There are online resources and tutorials available. Hence, it provides excellent documentation and support (1). Since it is an enterprise framework, the learning curve for Spring framework is quite high. You will need to be familiar with various APIs and its documentation. It introduces you to some new concepts that are different from normal programming paradigm. We have to learn new concepts such as dependency injection, Java servlets, and JSP, each of which is huge. Spring framework itself consists of a number of modules. Hence, the learning effort is quite high for Spring framework (5). The time taken to develop is also quite high. Spring is not suitable for the development of a small scale application. The development time will span up to more than a year using Spring framework. The number of lines of code required to achieve a certain functionality is much more in Java. As a result, the development time for Spring framework is higher than other frameworks (5).

Java code is divided into classes and packages. Since Spring uses Java it is mostly extensible. You can easily add new classes to Spring framework. But the integration to the framework is not quite easy (3). It is much more difficult to maintain a Spring application due to the complexity involved. As a result, the maintainability of Spring is satisfactory (4). Spring framework provides swing based user interface elements. These elements do not provide native look and feel of the environment. Since Java

is platform independent, the look and feel of the Spring framework is also platform independent. Third-party themes can be used to enhance the visual impact of the applications being developed, CSS and JavaScript. Hence the native look and feel is also satisfactory in Spring framework (5).

Spring MVC has a whopping size of 16 MB, therefore, is a heavy and complex framework with many components. This tends to reflect in its load time. As Java is heavy, clunky, and slow, Spring inherits these qualities from Java and tends to be slower than the other tested frameworks (5). The runtime performance of Spring is also poor. Since Spring uses Java servlets to run and serve web pages, its run-time performance is also slower (4).

14.6 Discussion

In this section, we summarize the weaknesses and strengths of each web framework. We will discuss scenarios and the scenarios in which different frameworks are suitable. Figure 14.3 illustrates the summary of our ranking for the analytical hierarchal process.

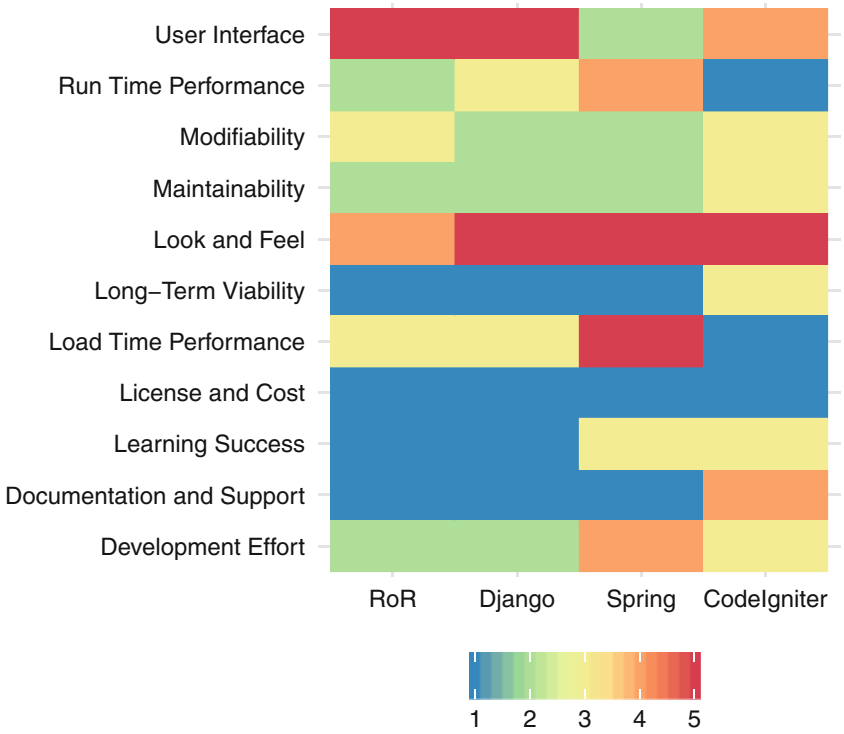

Fig. 14.3 Ranking of web frameworks against criterion on a scale of five

Ruby on Rails is a framework that is suitable for rapid application development. It is a popular software used by startup companies to develop web application due to ease of use and rapid development. Rails concentrates more on the business layer of application development. It provides advanced ORM functionalities. It would have been better if the user interface for Rails was defaulted to bootstrap. A Larger organization with huge applications will have to migrate if the total users and connections increase.

Django could also get some touch up on the UI front. The difference between ROR and Django boils down to the programming languages being used. Other than that, it is almost equivalent. Familiarity with Python or Ruby would be the decider in such scenarios.

CodeIgniter is written in PHP. PHP normally has an unsavory reputation of being spaghetti in style. But they produce ultra-fast, highly minimal applications with a minimal footprint. CodeIgniter can be used to develop applications that require faster response as it has an excellent loading time. It is suitable for smaller application. For larger application, it induces complexity and becomes an overhead.

Spring is an enterprise level framework that can handle data-intensive application. It is not suitable for developing a small application. For smaller applications it increases the development time and the complexity of the code, which degrades the performance of the smaller application, whereas for the larger enterprise solution, it is the best solution as it can handle data-intensive tasks and does handle heavy traffic and sizable connections without crashing the web server.

We have summarized the plus point and negatives of various scenarios, where the frameworks discussed can be used. We should have a weight for different criteria depending upon the requirements of the applications. Table 14.2 is from where you have to begin from where you can use additive principals to select an appropriate framework. In summary, if the web application is small and needs to be

Table 14.2 Results of the WebSelec for the example scenario showing the maximum and minimum for each criterion

	Weight	CodeIgniter	Ruby on Rails	Django	Spring
Select framework	100.00	**38.79**	26.23	20.38	14.60
Load-time performance	26.69	**18.46**	3.58	3.58	1.08
Real-time performance	24.16	**12.95**	6.68	2.99	1.55
Development effort	7.75	**0.77**	3.05	3.05	0.87
Long-term viability	7.35	**0.46**	2.30	2.30	2.30
Look and feel	6.78	1.13	**3.39**	1.13	1.13
Maintainability	6.29	**0.90**	1.80	1.80	1.80
License and cost	5.76	1.44	1.44	1.44	1.44
Documentation and support	5.29	**0.24**	1.68	1.68	1.68
Learning success	4.34	0.54	1.63	1.63	0.54
Modifiability	3.41	**1.15**	**0.54**	0.64	1.08
User interface	2.17	**0.74**	0.15	0.15	1.13

developed faster, then you have the option to select ROR or Django, depending upon the preferred programming languages. If the application is required to be fast, then CodeIgniter is a good option. If the application is being developed for a large organization with large development time frame, then Spring is a suitable framework.

14.7 Example Scenario

In this section, we discuss an example scenario wherein we use our SelecWeb tool for the selection of the best web framework that satisfies the requirements of an example application. The following were the selected requirements for the application in decreasing order of priority:

1. Load-time performance of the web application
2. Real-time performance of the web application
3. Modifiability of the application
4. User interface of the web application

Requirements such as look and feel, development effort, maintenance, and documentation were deemed as negligible or of lower priority. With these requirements, we use the SelecWeb tool to select the best format. The results of the SelecWeb tool are given in Fig. 14.4 and Table 14.2.

	Weight	CodeIgniter	Ruby on Rails	Django	Spring	Inconsistency
Select Framework	100.0%	38.8%	26.2%	20.4%	14.6%	25.6%
Load-Time Performance	26.7%	18.5%	3.6%	3.6%	1.1%	9.0%
Real-Time Performance	24.2%	12.9%	6.7%	3.0%	1.5%	0.8%
Developement Effort	7.7%	0.8%	3.1%	3.1%	0.9%	7.0%
Long-Term Viability	7.3%	0.5%	2.3%	2.3%	2.3%	0.0%
Look and Feel	6.8%	1.1%	3.4%	1.1%	1.1%	0.0%
Maintainability	6.3%	0.9%	1.8%	1.8%	1.8%	0.0%
License and Cost	5.8%	1.4%	1.4%	1.4%	1.4%	0.0%
Documentation and Support	5.3%	0.2%	1.7%	1.7%	1.7%	0.0%
Learning Success	4.3%	0.5%	1.6%	1.6%	0.5%	0.0%
Modifiability	3.4%	1.2%	0.5%	0.6%	1.1%	27.4%
User Interface	2.2%	0.7%	0.1%	0.1%	1.1%	10.1%

Fig. 14.4 The results of the WebSelec for the example scenario

In Fig. 14.4, **Weight** indicates the weight of each requirement that has contributed in the selection of appropriate web framework. We observe that with the given requirements AHP has given a higher score to CodeIgniter (38.8%), followed by Ruby on Rails (38.8%), and Django (38.8%). Higher score indicates a higher achievement of the provided requirements. Hence, with a higher score from our analysis using WebSelec, it is much more feasible to develop the web application using CodeIgniter for the achievement of the requirements. The bold, red values in Table 14.2 indicate the requirements with higher score and bold, blue colored values indicate the lowest score for a given requirement. Requirements such as load-time performance, real-time performance, and modifiability have higher weights for CodeIgniter (18.46% and 12.95%, respectively) as compared to other frameworks, whereas the requirements we assigned least priority such as development effort, long-term viability, maintainability, and documentation and support are lowest in CodeIgniter.

This example clearly illustrates the feasibility and utility of the Analytic Hierarchal Process for selecting a web application framework with multiple requirements or decision makers.

14.8 Conclusion

In this paper, we proposed SelecWeb, an automatic tool for selecting a web framework based on a set of criteria and developer preferences. We presented an analysis of web development framework. A set of criteria was derived, based on application requirements. The set of criteria was used to test and evaluate the web application frameworks. Ruby on Rails, Django, CodeIgniter, and Spring were the web application frameworks that were tested. Each framework was tested by the authors. The assessment and evaluation are valid for the near future but might change as the quality of technology varies. Hence the accuracy of the information is not guaranteed for longer time frame but the methodology and general information provided will remain applicable. Using a case study, we demonstrated the use of the SelecWeb tool to select the best web framework that satisfies the requirements of a given application.

Future work will focus on the security evaluation and detailed performance assessment of the web development frameworks. Moreover, the current development of the tool is based on the AHP method. In the future, we plan to use machine learning techniques to automatically predict the best web development framework for developers and users. The rankings of the web frameworks, based on the discussed criteria (license and cost, long-term viability, etc.), can be used to train a machine learning based model. The trained model will be able to automatically select the best framework based on the given preferences of the users and developers.

Acknowledgements The authors acknowledge with thanks the technical and financial support from the Deanship of Scientific Research (DSR) at the King Abdulaziz University (KAU), Jeddah, Saudi Arabia, under the grant number G-673-793-38. The work carried out in this paper is supported by the High Performance Computing Center at the King Abdulaziz University, Jeddah.

References

1. Muhammed, T., Mehmood, R., Albeshri, A., Katib, I.: UbeHealth: a personalized ubiquitous cloud and edge-enabled networked healthcare system for smart cities. IEEE Access **6**, 32258–32285 (2018)
2. Schlingensiepen, J., Nemtanu, F., Mehmood, R., McCluskey, L.: Autonomic Transport Management Systems—Enabler for Smart Cities, Personalized Medicine, Participation and Industry Grid/Industry 4.0., pp. 3–35. Springer, Cham (2016)
3. Mehmood, R., Graham, G.: Big data logistics: a health-care transport capacity sharing model. Procedia Comput. Sci. **64**, 1107–1114 (2015). Conference on ENTERprise Information Systems/International Conference on Project MANagement/Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN / HCist 2015 October 7–9, 2015
4. Mehmood, R., Meriton, R., Graham, G., Hennelly, P., Kumar, M.: Exploring the influence of big data on city transport operations: a Markovian approach. Int. J. Oper. Prod. Manag. **37**(1), 75–104 (2017)
5. Mehmood, R., Lu, J.A.: Computational Markovian analysis of large systems. Int. J. Manuf. Technol. Manag. **22**(6), 804–817 (2011)
6. Mehmood, R., Lu, J.A.: Computational Markovian analysis of large systems. Int. J. Manuf. Technol. Manag. **22**(6), 804–817 (2011)
7. Arfat, Y., Aqib, M., Mehmood, R., Albeshri, A., Katib, I., Albogami, N., Alzahrani, A.: Enabling smarter societies through mobile big data fogs and clouds. Procedia Comput. Sci. **109**, 1128–1133 (2017). 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16–19 May 2017, Madeira, Portugal
8. Suma, S., Mehmood, R., Albugami, N., Katib, I., Albeshri, A.: Enabling next generation logistics and planning for smarter societies. Procedia Comput. Sci. **109**, 1122–1127 (2017). 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16–19 May 2017, Madeira, Portugal
9. Arfat, Y., Mehmood, R., Albeshri, A.: Parallel shortest path graph computations of United States road network data on apache spark. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 323–336. Springer, Cham (2018)
10. Alam, F., Mehmood, R., Katib, I.: D2TFRS: an object recognition method for autonomous vehicles based on RGB and spatial values of pixels. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 155–168. Springer, Cham (2018)
11. Muhammed, T., Mehmood, R., Albeshri, A.: Enabling reliable and resilient IoT based smart city applications. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 169–184. Springer, Cham (2018)
12. Alotaibi, S., Mehmood, R.: Big data enabled healthcare supply chain management: opportunities and challenges. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 207–215. Springer, Cham (2018)
13. Khanum, A., Alvi, A., Mehmood, R.: Towards a semantically enriched computational intelligence (SECI) framework for smart farming. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 247–257. Springer, Cham (2018)

14. Usman, S., Mehmood, R., Katib, I.: Big data and HPC convergence: the cutting edge and outlook. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 11–26. Springer, Cham (2018)
15. Alomari, E., Mehmood, R.: Analysis of tweets in Arabic language for detection of road traffic conditions. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 98–110. Springer, Cham (2018)
16. Suma, S., Mehmood, R., Albeshri, A.: Automatic event detection in smart cities using big data analytics. In: Mehmood, R., Bhaduri, B., Katib, I., Chlamtac, I. (eds.) Smart Societies, Infrastructure, Technologies and Applications, pp. 111–122. Springer, Cham (2018)
17. Mehmood, R., Alam, F., Albogami, N.N., Katib, I., Albeshri, A., Altowaijri, S.M.: UTiLearn: a personalised ubiquitous teaching and learning system for smart societies. IEEE Access **5**, 2615–2635 (2017)
18. W3C: HTML5. http://www.w3.org/TR/html5/ (2015). Online; Accessed 20 May 2015
19. Bachle, M., Kirchberg, P.: Ruby on rails. IEEE Softw. **24**(6), 105–108 (Nov 2007)
20. Arthur, J., Azadegan, S.: Spring framework for rapid open source J2EE web application development: a case study. In: Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005, pp. 90–95. IEEE, Piscataway (May 2005)
21. Raible, M.: My comparing JVM web frameworks presentation from Devoxx 2010. https://raibledesigns.com/rd/entry/video_of_comparing_jvm_web (2013). Online; Accessed 20 May 2015
22. Rodriguez-Martinez, M., Seguel, J., Greer, M.: Open source cloud computing tools: a case study with a weather application. 2013 IEEE Sixth Int. Conf. Cloud Comput. **0**, 443–449 (2010)
23. Smutny, P.: Mobile development tools and cross-platform solutions. In: 2012 13th International Carpathian Control Conference (ICCC), pp. 653–656. IEEE, Piscataway (May 2012)
24. Heitkötter, H., Majchrzak, T., Ruland, B., Weber, T.: Comparison of mobile web frameworks. In: Krempels, K.H., Stocker, A. (eds.) Web Information Systems and Technologies. Lecture Notes in Business Information Processing, vol. 189, pp. 119–137. Springer, Berlin (2014)
25. Wikipedia: Web application framework — Wikipedia, the free encyclopedia (2015). Online; Accessed 20 May 2015
26. Python Community: WebFrameworks - Python Wiki. https://wiki.python.org/moin/WebFrameworks (2018). Online; Accessed 20 May 2018
27. Ruby, S., Thomas, D., Hansson, D.H.: Agile Web Development with Rails. Pragmatic Bookshelf (2011). ISBN: 1934356549, 9781934356548
28. Wikipedia: CodeIgniter—Wikipedia, the free encyclopedia (2013). Online; Accessed 20 May 2015
29. Stack Overflow: What is dependency injection? http://stackoverflow.com/questions/130794/what-is-dependency-injection (2008). Online; Accessed 20 May 2015
30. Programmers Stack Exchange: Spring introduction and research. http://programmers.stackexchange.com/questions/92393/what-does-the-spring-framework-do-should-i-use-it-why-or-why-not (2011). Online; Accessed 20 May 2015
31. Saaty, T.L.: Axiomatic foundation of the analytic hierarchy process. Manage. Sci. **32**(7), 841–855 (July 1986)
32. Apache: Apache license, version 2.0. http://www.apache.org/licenses/LICENSE-2.0 (2018). Online; Accessed 20-May-2015
33. MIT: The MIT license (MIT)-Open Source Initiative. http://opensource.org/licenses/MIT (2018). Online; Accessed 20 May 2015
34. Sen, R., Subramaniam, C., Nelson, M.L.: Open source software licenses: strong-copyleft, non-copyleft, or somewhere in between? Decis. Support Syst. **52**(1), 199–206 (December 2011)
35. GNU Foundation: Licenses - GNU project - Free Software Foundation. http://www.gnu.org/licenses/ (2018). Online; Accessed 20 May 2015