Assignment 5.

Ruby on Rails(web application)

Roll no. 201301015 SEFP
Manish Tanwar Instructor- Sridhar sir,

Abstract. –

Ruby on Rails is a web development framework for database back-ended applications. This framework is becoming very popular in the agile development due to its productivity and agility features. Ruby on Rails have built-in solutions to many common problems that developer face during web development. Furthermore, this framework favors convention over configuration, which makes the web development more agile. Additionally, these conventions make the web applications more maintainable as well, because the developers adopt the Rails convention and sensitive defaults.

1. What Is Ruby on Rails?

Ruby on Rails (RoR) is open source web framework written in the Ruby programming language, and all the applications in Rails are written in Ruby. Ruby on Rails is focused on productivity and enforces agile web development.

It's a dynamically typed, fully object-oriented, general-purpose scripting language. Ruby was designed to have an elegant syntax and made as human readable as possible, for instance it does not need colons and parenthesis around parameters. Some parts of the code are read like English declarations.

2 . Funcional and nonfunctional Features of Roby on Rails:-

- Model-View-Controller architecture.
- Representational State Transfer (REST) for web services.
- Supports the major databases (MySQL, Oracle, MS SQL Server, PostgreSQL, IBM DB2, and more).
- Open-source server side scripting language.
- Convention over configuration
- Scripts generators to automate tasks.
- Use of YAML machine, which is a human-readable data serialization format.
- The above-described features are distributed in the following Rails' components and the Fig. 2 shows the interaction between some of these components:

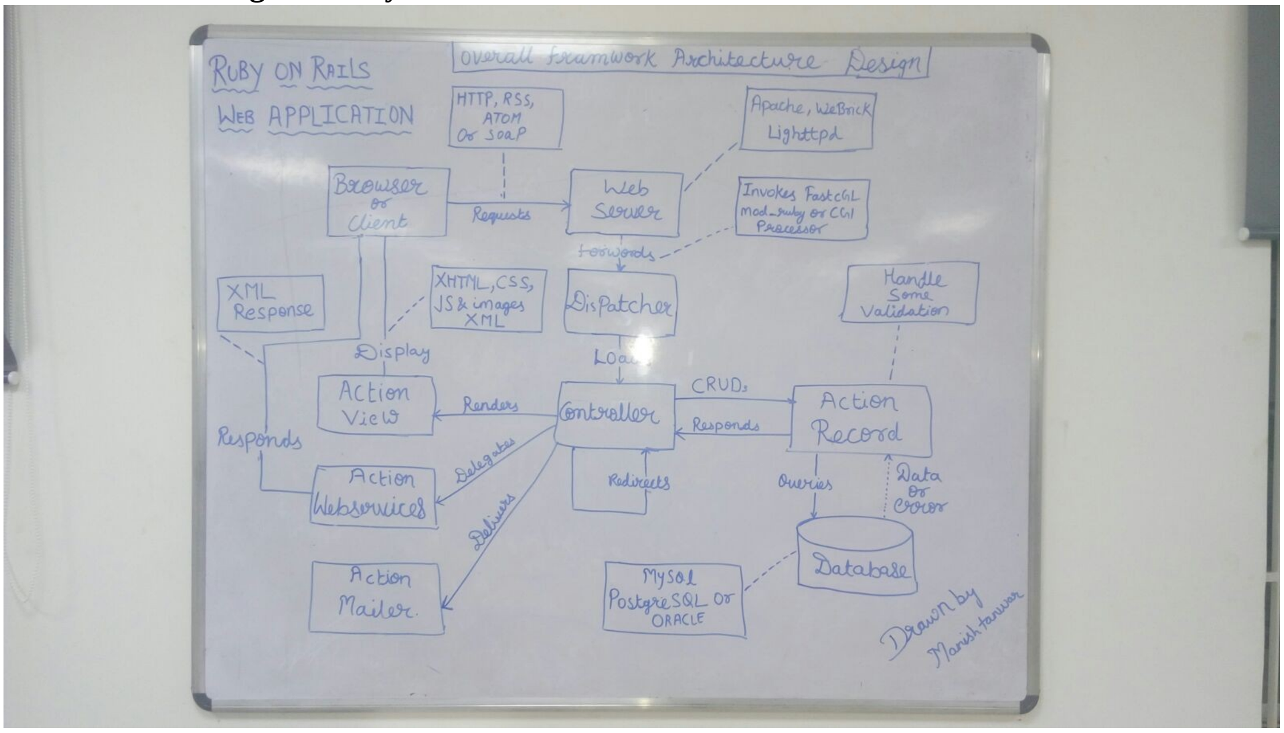
- Action Mailer
- Action Pack
- Action Controller
- Action Dispatcher
- Action View
- Active Model
- Active Record
- Active Resource

- Active Support
- Railties

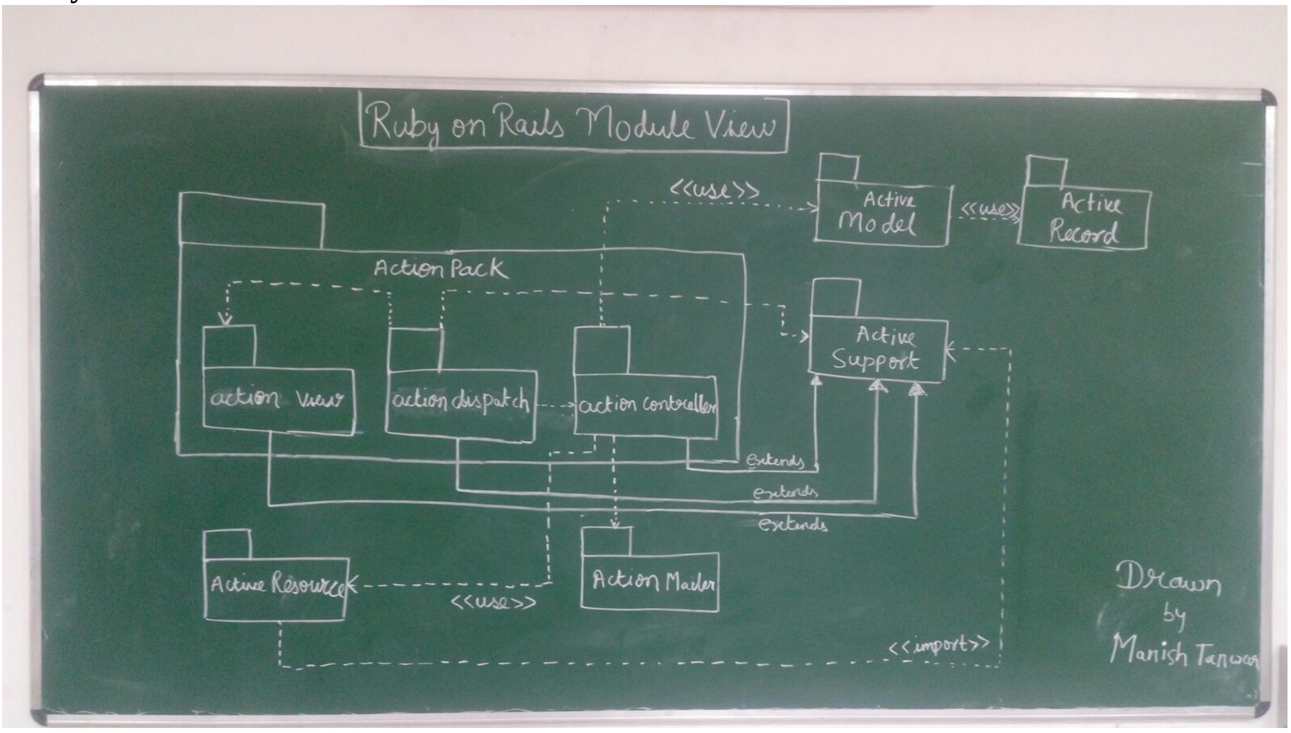
3. Ruby on Rails Architectural Design:-

The Ruby on Rails framework was designed for database-backed web applications. It was created as a response to heavy web frameworks such as J2EE and the .NET framework. In order to make the development process faster, Ruby on Rails uses conventions and assumptions that are considered best ways to accomplish tasks, and it's designed to encourage those. This convention eliminates configuration code and increases productivity. Many of the common tasks for web development are built-in in the framework to work out-of-the-box. This includes email management, object-database mappers, file structures, code generation, how the elements are named and organized and so on. All of these conventions allow developers to write less code and develop agile applications. Additionally, the enhance maintainability and understandability around the Ruby on Rails developers' community.

Architecture Design of Ruby on Rails.



Ruby on Rails Module view.



4. Explaination of Design process and Architecture in detail:-

Model-View-Controller Pattern

Ruby on Rails uses the Model-View-Controller (MVC) architectural pattern in order to improve the maintainability of the application. The Model centralizes the business logic, the View manages the display logic, while the Controller deals with the application flow. The MVC allows a clean separation of concerns, in the way that it keeps the business logic separated from HTML views. Additionally, it improves decoupling and testing.

* **Model**

The Model layer carries the business logic of the application and the rules to manipulate the data. In Ruby on Rails, the models are used to manage the interaction with their corresponding elements in the database. The Models represent the information in the database and do the appropriate validations.

* **View**

The view is the front-end of the application, representing the user interface. In Ruby on Rails, views are HTML files with embedded Ruby code. The embedded Ruby code in the HTMLs is fairly simple (loops and conditionals). It is only used to display data to the user in the form of views. Views are used to provide the data to the browsers that requested the web pages. Views can server content in several formats, such as HTML, PDF, XML, RSS and more.

* **Controller**

Controllers interact with models and views. The incoming requests from the browsers are processed by the controllers, which process the data from the models and pass it to the views for presentation.

Rails Modules

* **Action Mailer**

This module is responsible for providing e-mail services. It processes incoming mails and creates new ones. This module can handle simple text or complex rich-format emails. Also it has common tasks built-in, such as, sending out forgotten passwords, welcome messages, and fulfilling any other written-communication’s need. Action Mailer is wrapped around the Action Controller. It provides ways to make email with templates in the same way that Action View uses it to render web pages.

* **Action Pack**

The Action Pack module provides the controller and view layers of the MVC patterns. These modules capture the user requests made by the browser and map these requests to actions. These actions are defined in the controllers layer and later the actions render a view that is displayed in the browser. Action Pack is divided in 3 sub-modules, which are: Action Dispatch, Action Controller, and Action View.

Action Dispatch:
Handles routing of web browser request. It parses the web request and does

advanced processing around HTTP, such as handling cookies, sessions, request methods, and so forth.

Action Controller:

After the action dispatch has processed the request it makes the routing to its corresponding controller. This module provides a base controller from which all the other controllers can inherit. Action Controller contains actions to controls model and view. This module makes data available as needed, controls views rendering and redirection. Additionally, it manages the user sessions, application flow, caching features, helper modules and implement filters for the pre, during and post processing hooks.

Action View:

It is call by the Action Controller. It renders the presentation of the web page requested. Action View provides master layouts, templates lookups and view helpers that assist the generation of the HTML, feeds and other presentation formats. There are three templates schemas in Rails, which are rhtml, rxml, and rjs. The rhtml format generates HTML views to the users with ERB (embedded ruby code in HTML). The rxml is used to construct XML documents using Ruby, and rjs allow creating dynamic JavaScript code in Ruby useful to implement AJAX functionality.

RESTful Architecture of Roby on Rails:-

REST stands for Representational State Transfer. REST is an alternative to web services, such as SOAP and WSDL. It relies in the HTTP protocol for all the CRUD operations: create, read, update and delete. RESTful web services are appropriated when the web services are completely stateless, limited bandwidth (it's very useful for mobile devices since it doesn't the the overhead of other protocols like SOAP), when the data is not generated dynamically so it could be cached to improve performance and when there is a mutual understanding between the service producer and the consumer.

Pitfalls of the Ruby on Rails Architecture:-

Since Rails is built on the Ruby language it inherits the goodness and weakness of that language. Ruby is a dynamic scripting language with an elegant syntax and fully object-oriented. Because it is an interpreted language it is slower than other languages that are compiled like Java or C++. In most cases, this difference in speed is not a problem but when the web application needs to scale to millions of concurrent users this performance starts to degrade. The Ruby language is not suited for high concurrency applications, because it is not optimized for speed computing. It was designed to be elegant, simply and for rapid development. Ruby doesn't have good thread support yet and, like many other scripting languages, Ruby has trouble dealing with long-lived processes. But other languages, like Java, are really good at that because they have been optimized for year to handle threads efficiently. Another weakness is the Ruby's garbage collector is not as good as Java's in that each process requires much more memory. In terms of deployment, a web application on Ruby on Rails could be harder to deploy than sites that are using more common technologies, such as PHP. That's because not all the hosting providers support Rails, but in time the support of Rails is increasing.

5. Alternative design solution :-

The selected views to analyze this architecture are Module Views and Component and Connector (C&C) Views. The Module View will contain UML diagrams which represent a static view of all the components, while the Component & Connector View will show the UML diagram that presents a run-time view of a system's architecture: what components exist at run-time and how do these components communicate with one another.

6 Summary

In this complete design process description the main idea is to make understand how this web application works and how its architecture designed in such way that it is userfriendly and useful Ruby on Rails has received widespread support throughout the software development industry, more specifically the open-source community. This support reflects, to a certain extent, that the framework was able to provide a fairly robust architecture somewhat consistent with the goals that the architect envisioned.