Cloud-based Project Management System

Dr. R. Kavitha, Kishan Lal L S, Aziz Rahman R

[1]Professor, Department of Information Technology, Velammal College of Engineering & Technology, Madurai, India
[2]Student, Department of Information Technology, Velammal College of Engineering & Technology, Madurai, India
[3]Student, Department of Information Technology, Velammal College of Engineering & Technology, Madurai, India

---***---

Abstract – *The success of a project is all about finding a great idea for managing the projects. Effective management and technical support are required for the success of a large-scale project. Managing and storing such large-scale projects and their relevant details such as assigned employees, client information, skill-set requirements in one place for a longer period with easy accessing and manipulation of the same is a tedious process. We proposed a cloud-based application that aims to eliminate these difficulties by providing a platform to read, store, and manage projects' details with high optimization algorithms and user-friendly/minimalist UI/UX with more security levels. By using this application, it'll be easier to plan projects while taking previous track records into account. Additionally, we can make sure that people are working on the right things at the right time. As the project teams are getting remote and larger, better collaboration is required and our application, which is a cloud-based technology, facilitates this seamless collaboration around the globe in a click.*

Key Words: Project Management, Cloud, AWS, Serverless, Angular, Spring Boot

1. INTRODUCTION

Project Management has been revolving around the world for centuries. From Seven Wonders of the World to today's cloud applications, the need for successful project management, not just for a status symbol but it's becoming the essential pillar of strength and support to get the job done. Be it a small-scale business, decentralized team, or a worldwide brand, there should be a channel for managing the projects. During the inception phase of a project, every ambitious entrepreneur develops high hopes and rewarding results. It is a true struggle to ascertain the project growing from start to finished product. When taking into the picture, the project management software, people think of it as a method of managing the tasks required to fulfill a project, within the timeframes set, to achieve the desired outcome.

The origin of the modern project management concept started between the 1900s and 1950s. During this period, technology advancement shortened the project schedule. Automobiles allowed effective resource allocation and mobility. Telecommunication systems increased the speed of communication. The job specification was widely used and Henry Gantt invented the Gantt chart.

The benefits of project management software are boundless. Earlier, project management software was symbolized with a pencil, a sheet of paper, and a strategy/methodology. However, time marches on and technology evolves. Essentially, project management software programs are employed for project planning, time management, resource allocation, etc. Project management software is among the foremost useful tools that will help companies in becoming as competitive as possible and set them apart from the rest of the industry.

1.1 Aim

The main objective of this cloud-based application is to deliver a centralized system for managing the projects' details and their status, client assigned, required skill-set, and resource allocations in an effective and efficient way.

1.2 Overview

This application helps to manage the project details, resource allocation, client, skill-set requirements to keep track of everything and avoids ending up going out of control. The application can be accessed by two levels of users - Manager, and resource. The manager is the one who has the ability to edit the project details such as the name of the project, duration of the project, add/remove clients to/from the project, add skills needed for the project, assign/remove resources to/from the project, and delete the project. Allocation of resources to a project can be done based on the skill-set that a resource possesses and the skills required by the project. In this way, a manager can easily identify the suitable resource for a project based on the skills required by the project. Managers can also be able to view the resource information such as his /her names, availability of the resource, years of experience, and the resume of the resources, remove/add projects to the resource. With respect to clients, managers can also be able to add/remove clients to a project (single client per project), edit the name of the client, toggle the client status among active/inactive, and remove the client. When it comes to skill-set, managers can add a new skill, update existing skills, approve/reject skill requests from resources and delete a skill.

2. SYSTEM ANALYSIS

2.1 Existing System

The major concern with some of the existing systems is **Security**. As the Project Management System requires some of the personal details of the employees and employers to be stored, some of the existing system, which still merely depends on the excel sheet to manage the projects, fails to provide a guarantee over security. It's easy for attackers to get all the confidential information of the existing application as they use modern technologies for hacking, whereas the existing system still uses traditional old technologies.

Since the existing system is still being developed with old traditional programming languages, it sometimes fails to be **compatible with the recent technologies** and thus fails to meet the requirements and expectations of the customers who need their applications to be integrated with modern technologies. As a result of using traditional technologies, storing the information of thousands of employees and employers becomes inefficient as it doesn't have any optimization techniques.

The existing systems are **not scalable** (scalable - the ability to quickly and easily increase or decrease the resources). As a result, existing systems fail to maintain or increase the performance (i.e. profits, margins) even when faced with larger operational demands (increased sales).

In the existing system, we have to allocate some storage for the **data backup** and configure it manually which requires some **cost and time**. The application might crash during the production stage. At that time, we have to have some automatic backup feature to recover from the crash which I feel is still missing in most of the existing systems.

2.2 Proposed System

To overcome the limitations in the existing system, our application contains the following features. It uses **Two-Factor authentication**, a security system that requires two distinct forms of identification in order to access any resource inside the application. Two-factor authentication (2FA) does this by requiring two types of information from the user—a **password** or **personal identification number (PIN)**, a **code** sent to the user's smartphone, or a fingerprint—before whatever is being secured can be accessed. Moreover, it uses Microsoft's **Active Directory** that allows network administrators to create and manage domains, users, and objects within a network. Active Directory provides **single-sign-on (SSO)** to authenticate a user in multiple web applications in a single session. Thus, a user doesn't have to create an application-specific account to login into the application. Rather, they can use the account created for them in their organization. From a development perspective, our application uses AWS services such as EC2, S3, Lambda, RDS, etc. which are **highly scalable** and supports **on-demand provisioning**, which avoids overutilization and underutilization of resources. The application is developed using **Angular**, a front-end framework developed by Google. Angular is a **Component-based architecture** that provides a higher quality of code. It uses **TypeScript** which helps to enable better tooling, write cleaner code, and provides higher scalability. In addition to this, Angular uses **Hierarchical dependency injection**, which provides high-performance scores for any Angular application. The application uses **Serverless** framework as one of the backend frameworks, which can auto-scale and charge only when used. It's easily coupled with most of the AWS services such as AWS Lambda, AWS API Gateway, etc., and can be written in any of the popular programming languages such as Python, Java, NodeJs, Ruby, Go, C#, etc.

3. SYSTEM DESCRIPTION

The application has two levels of users: **Manager** and **Resource. Fig -1** clearly depicts the flow of each user and the shared flow as well.

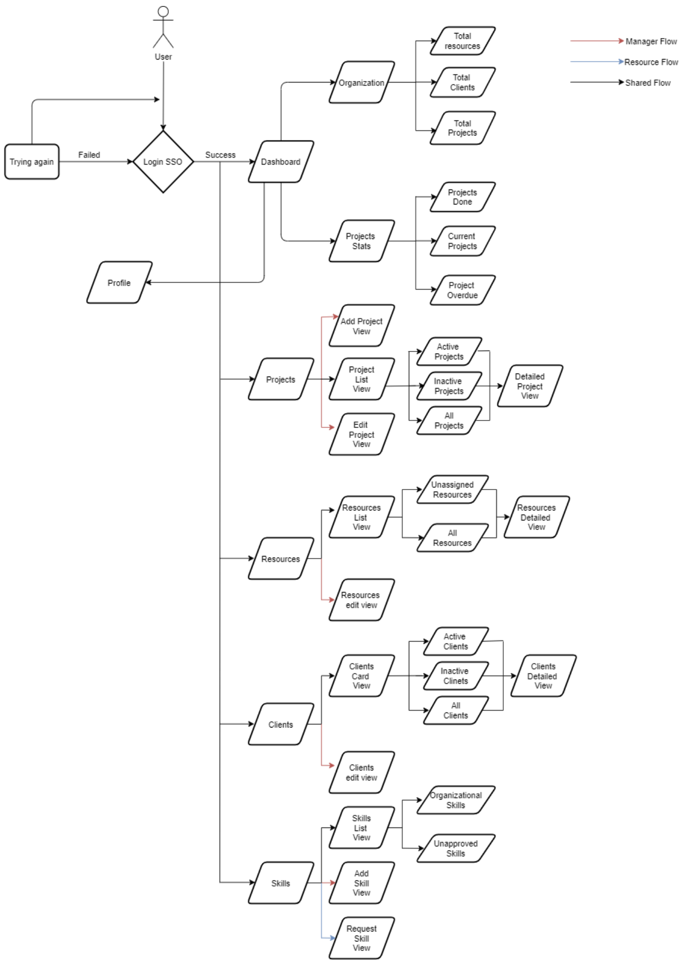

Fig -1: Flow Diagram

The Manager will have access to the following screens:

Dashboard - This view has projects stats, total counts of clients, resources and projects.

Project View – This page has a list of projects that the company is working on. On clicking on any of the projects, the manager will be taken to a detailed view of the project that has all the information collected when adding a project along with resources names and ability to add a resource to the project, and skill-set required for the project.

Resources List - This view is having a list of resources, their current project, their skills, and the date from when the resource is available for a new project. And they can filter resources that aren't yet assigned to any projects. On clicking on any of the resources, the manager will be taken to a detailed view of resources that the resource's details such as his/her name, skill set, current project, past projects, resume, interests, and so on.

Client view - This view shows a list of clients in a card view. On clicking on any of the resources, the manager will be taken to a detailed view of the client that has the name of the client, their logo, and the projects they are working on, and so on.

Skillsets view - This view shows a list of skills, their descriptions, and a total number of resources that possess the skill. The manager can also see the skills that are requested by the resource to add them to the main skill sets list.

Profile – This view contains the details of the logged-in manager. Here, they can edit their personal skill set, upload their resume and they can see the projects they are working on, and so on.

Manager Actions:

1. Manager will be able to add a new project with the following details:
    a. Project Name
    b. Short description
    c. Client details
    d. Skillset requirements
    e. Start and End date
2. Manager will be able to add resources to a project, and download the resume of any resources.
3. Manager will be able to add skill lookup and approve or reject skills requested by the resource.
4. Manager will be able to edit their personal information.

The Resources are the ones who don't have much access to the application as the managers. They are the ones with the kind of read-only/restricted access to the application and will have access to the following screens:

Dashboard - This view has projects stats, total counts of clients, resources and projects.

Project View – Resources are only allowed to have a look at the total projects that the company is working on and the detailed view of any project, but unlike managers, they aren't able to edit or delete the project.

Resources List – Resources have access to the resources list and detailed view, but can't modify the data. And they're not allowed to download any of the resources' resumes but are able to download their own resume from the list of resources.

Client view – Client card view and detailed view are the pages that the resources have access to, but can't edit/delete any data in those pages.

Skillsets view – Resources are allowed to access the skill sets list view and they can request the manager to add a new skill that arrived in the market so that it appears in the skill-set list view. Skills that are requested but aren't approved by managers are appeared in the "Unapproved skill sets list" view.

Profile – This view contains the details of the logged-in resource. Here, they can edit their personal skill set, upload their resume and they can see the projects they are working on, and so on.

Resources Actions:

1. Resource will be able to add their skills - (can be selected from pre-determined list added by manager)
2. Resource will be able to add their resume to their profile.
3. Resource will be able to edit their personal information.

4. SYSTEM ARCHITECTURE

The frontend of the application is built with angular and hosted in an EC2 instance. The backend is built with spring boot and AWS lambda with the serverless framework. The API requests are handled by the API gateway which interacts with the lambda functions to provide the data. The database (MySQL) is hosted in a separate RDS instance where all the data will be stored and retrieved. The user files are stored in the S3 bucket via a signed URL and are fetched directly when needed. The user identities are managed by the Azure active directory which is integrated with AWS Cognito to provide SSO, MFA for sign-in by users. **Fig -2** illustrates the architecture of this application.

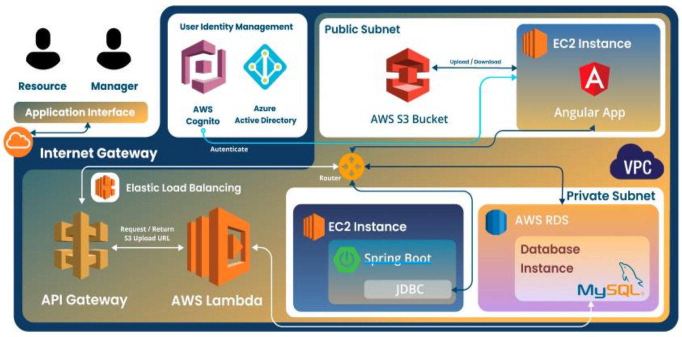

Fig -2: Architecture Diagram

Angular – Angular is a front-end framework used for creating efficient and sophisticated single-page apps.

Spring Boot - Spring Boot is an open-source Java-based framework for creating micro-services.

AWS Lambda - AWS Lambda is a serverless computing service that triggers and runs the code in response to certain events and also self-manages the computing resources required by that code.

AWS Cognito - Amazon Cognito is a web-service that provides user identity and data synchronization that helps to securely manage and synchronize app data for users across their devices.

Active Directory - Active Directory stores information about objects on the network and makes this information easy for administrators and users to seek out and use.

AWS RDS - RDS is distributed Relational Database Service provided by Amazon Web Services. It is a web service designed to simplify the setup, operation, and scaling of databases for use in applications.

AWS EC2 - Elastic Compute Cloud (EC2) is a web service that provides complete control of secure computing resources and lets us run applications in a proven computing environment.

AWS S3 - Amazon Simple Storage Service (S3) is a storage service used to store and retrieve data, at any time, from anywhere on the web.

AWS API Gateway - Amazon API Gateway simplifies the process of creating, managing, and securing APIs at any scale.

AWS Elastic Load Balancing - Elastic Load Balancing helps to automatically manage the traffic between multiple cloud resources to increase availability and to reduce the load.

MySQL - MySQL is a popular relational database used for managing relational databases.

5. FUTURE WORK

5.1 AWS Load Balancers

AWS Elastic load balancers automatically distribute incoming application traffic across multiple Amazon EC2 instances. It enables us to achieved increased levels of fault tolerance for our application by seamlessly providing the required amount of load balancing capacity needed to distribute the application traffic.

5.2 Swagger UI

Swagger is an excellent tool for documenting APIs. The most popular way of documenting APIs is known as Swagger UI. We can make use of this to document our spring boot APIs which ease our work of getting API information such as the API endpoints, request methods, response body format, payload structures, response codes, error responses, and their codes, etc.

5.3 Fetching Profile picture from Azure AD

We can make use of Microsoft's Graph API to fetch the profile picture of the users from their AD accounts.
https://docs.microsoft.com/en-us/graph/api/profilephoto-get?view=graph-rest-beta

5.4 Lambda layers

Lambda Layers enables you to centrally manage code and data that is shared across multiple functions and the Lambda Runtime API provides a simple interface to use any programming language or specific language version for authoring your functions. We can use Lambda Layers to enable re-use and sharing of code and can build and test Layers locally using the AWS Serverless Application Model (SAM).

5.5 AWS Secrets Manager

It's not recommended to keep all the confidential information, such as user/DB credentials inside our code or even in a separate file with the environmental variables. For that, we can use AWS Secrets Manager which helps us protect secrets needed in our application. It enables us to easily rotate, manage, and retrieve database credentials. We can retrieve the secrets from Secrets Manager with a call to Secrets Manager API. Additionally, it enables us to control access to secrets using fine-grained permissions and audit secret rotation centrally for resources in the AWS Cloud.

6. CONCLUSION

This cloud-based project management system helps to detail all the information about the projects. It helps us to maintain and track information like the resources and time involved in the projects. Project management software is a crucial tool in developing a project and thus every company should implement it in its system to increase productivity.

REFERENCES

[1]. M. Roberts, "Serverless Architectures," MartinFowler.com, May 22, 2018. [Online]. Available: https://martinfowler.com/articles/serverless.html

[2]. Amazon Web Services, "Serverless Computing and Applications," AWS, 2018. [Online]. Available: https://aws.amazon.com/serverless

[3]. Manoj Kumar, "Serverless Architectures Review, Future Trend and the Solutions to Open Problems", January 25, 2019, American Journal of Software Engineering. [Online]. Available: http://pubs.sciepub.com/ajse/6/1/1/

[4]. Mubbashir Mustafa, "Integrate Azure Active Directory with AWS Cognito User Pool", Nov 23, 2020. [Online]. Available: https://dev.to/mubbashir10/integrate-azure-active-directory-ad-with-aws-cognito-user-pool-1l7h

[5]. Rutvik Thakkar, "Create API using ExpressJS and Sequelize", September 26, 2019. [Online]. Available: https://blog.devitpl.com/sequelize/

[6]. Ryan Zhou, "Using AWS RDS for your Spring Boot App", July 10, 2018. [Online]. Available: https://medium.com/@ryanzhou7/using-aws-rds-for-your-spring-boot-app-ca8f4b09c9b8

[7]. Kumar Gaurav, "Deploying/Hosting Spring boot applications on AWS EC2", October 2, 2020. [Online]. Available: https://medium.com/@kgaurav23/deploying-hosting-spring-boot-applications-on-aws-ec2-7babc15a1ab6

[8]. Sandeep Tengale, "Deploy Angular App on AWS EC2 instance", April 19, 2019. [Online]. Available: https://medium.com/@sandeeptengale/deploy-angular-app-on-aws-ec2-instance-20749f17b33e

[9]. Shri Ganesh Hegde, "Postman tool: Simplifying and Reshaping API testing", June 17, 2019. [Online]. Available: https://www.specbee.com/blogs/postman-tool-simplifying-and-reshaping-api-testing