2019 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)

Cloud Based Algorithm for Task Management

Parvizsho Amnov, Navjot Bola, Dipti Shiralkar, Meghana Yoganarasimha
College of Engineering, San Jose State University
One Washington Square San Jose, CA 95192 USA
parvizsho.aminov@sjsu.edu
navjot.bola@sjsu.edu
dipti.shiralkar@sjsu.edu
meghana.yoganarasimha@sjsu.edu

***Abstract*— Time management is challenging because not only does one want to ensure they are able to get work done on time, they also want to utilize time in the most efficient way. This paper proposes a solution developed using Cloud Computing and a scheduling algorithm based on Pomodoro technique.**

***Keywords*— Cloud Computing, Serverless Architecture, Time management, Pomodoro technique.**

I. Introduction

Cloud Computing (CC) is very popular in today's world, as it has helped the industry evolve by utilizing faster delivery of solutions for many day-to-day problems. One of such problems is time management, a task that can be an extremely tedious. Nowadays, there are many methodologies of time management that folks honor for their success. Unfortunately, many of these techniques of time management do not always enable one to maximize their time and effort. The Pomodoro technique [1], on the other hand, provides a process for improving productivity which helps enhance ones focus and concentration [2], boost motivation, and improve the overall work/ study process [3][4].

Pomodoro technique is a time management method developed by Francesco Cirillo. The pomodoro technique methodology breaks down tasks into compact pieces that can be performed in smaller time intervals (usually about 25 minutes) separated by short breaks. These task intervals are called

This paper goes over the detailed cloud architecture of PomaFocus along with the scheduling algorithm. Identifying the cloud architecture helps to understand how to design such a solution and make it cost effective and scalable. The real complexity of the solution is in its scheduling algorithm, which divides the user task into a number of pomodoros. It does this by leveraging machine learning and automatically scheduling and rescheduling a day for the user. The scheduling is done based on user preferred time slots, pomodoros derived from the pending tasks, and user's progress.

II. Proposed solution

PomaFocus is a task management solution based on Pomodoro technique, which allows users to optimize their day. It is designed using Cloud Architecture and it utilizes cloud based Machine learning services. The most noteworthy aspect of PomaFocus is task scheduling. A user simply needs to enter their tasks, a priority (if desired), and a daily schedule is generated for them. The integration with personal calendars is an important feature of PomaFocus, which ensures scheduling of pomodoros around existing meetings is possible.

A. Features

PomaFocus gives users the ability to:

- Create and manage projects and tasks
- Have their days scheduled automatically
- Send notifications when a task should start
- Track pomodoro cycle with in-app timer
- Automatic rescheduling of tasks if task has not been completed
- Integration with personal calendar
- View task analytics and data

978-1-7281-1664-8/19/$31.00 ©2019 IEEE
DOI 10.1109/CSE/EUC.2019.00055

III. ARCHITECTURE

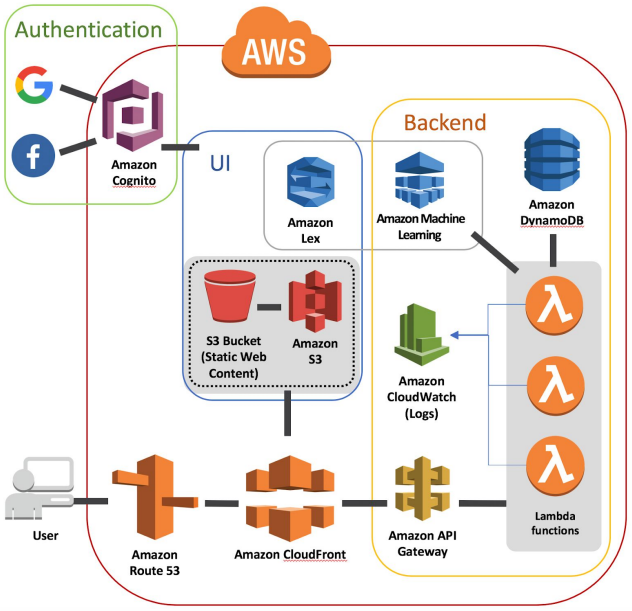

Fig. 1 A serverless web application architecture designed for PomaFocus

PomaFocus utilizes serverless framework [5] with a 3-tier architecture as shown in Fig. 1. The proof of concept (POC) architecture uses all AWS services; however, it can be similarly deployed on any cloud provider. Serverless Computing is emerging as a new and compelling paradigm for the deployment of cloud applications [6]. The main advantage it provides is that the resources are used only when needed, as Lambda functions will be executed only when called. This decreases operational costs. The design of PomaFocus solution ensures that scheduler is executed only when User adds new task or reschedule is triggered. This means scheduler process need not run always and hence serverless architecture can be used to make the solution cost effective and efficient.

Another important Architectural consideration is the database (DB). The PomaFocus solution leverages a database to store user preferences, projects, tasks, and schedule. AWS DynamoDB was chosen as part of POC implementation, considering the advantages of scalability and security. However, while implementing the solution it was observed that the database tables do have necessary relations and relational database could be a better fit for this solution.

All the create, update, delete (CRUD) operations on DB as well as all actions around scheduling or rescheduling are developed as lambda functions and deployed as REST API endpoints at AWS Cloud Gateway. Complete solution is deployed using Cloud formation template. This architecture makes it easy to manage and maintain the application. Any fix can be seamlessly redeploy REST API lambda function within minutes.

The user interface (UI) and user experience (UX) considerations are important and are discussed in detail in section IV. UI contains pages required for different user activities like configuration, adding a project and tasks, calendar display, integration with other calendars. Also, UI provides interactive notifications upon completion of a pomodoro. The static UI content is deployed and serviced from AWS S3 bucket service. In order to improve user experience PomaFocus is using interactive Bot that helps collect and configure user's account.

The most important task provided by the system is scheduling and rescheduling tasks. Scheduling is performed when a user adds new project/task and at the time pomodoro completion. Additionally, the tasks are rescheduled whenever a user makes changes to the integrated calendar. Scheduling and rescheduling of tasks only occurs when user performs certain triggers. Therefore, the use of Serverless design (i.e. lambda services provided by AWS) is beneficial as functions are executed on need basis. This algorithm is discussed in detail in the next section. It also discusses how algorithms uses machine learning service to predict the number of pomodoros required to complete a task and thus help users plan ahead.

Last architectural consideration is the performance of the system. To improve the performance and end-user experience, UI and backend APIs are configured via CloudFormation and served through CloudFront to accelerate the performance by

bringing all functionality to the edge location near to application users.

A. Scheduling Algorithm

The most important component provided by PomaFocus system is scheduling and rescheduling of user's tasks. The scheduling algorithm takes care of two actions: create new schedule and reschedule events. The algorithm flow for scheduling is depicted in Fig. 2 and rescheduling in Fig. 3. Both actions rely on user provided information about their work day, the projects, and the tasks.

Scheduling is done by following a few basic steps:

1) Find empty time slots in user's existing calendar.
2) Analyze how many pomodoros can fit into the timeslot (including breaks).
3) Optionally use Machine Learning to predict the number of pomodoros it will take to complete each task and utilize the result when building schedule.
4) Create entries in database for each of the pomodoros which include start and end times.

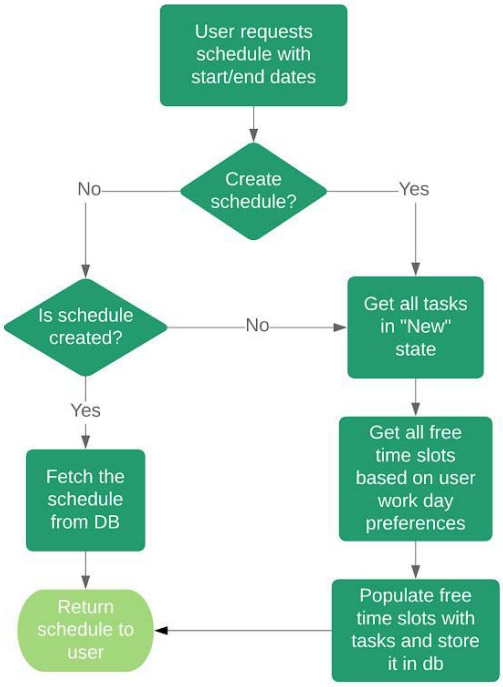

Fig. 2 Scheduling flowchart

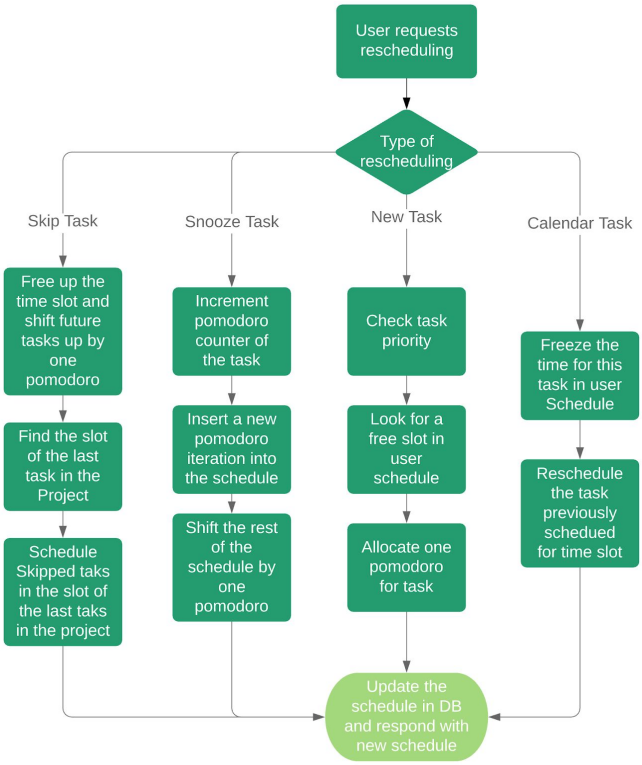

Fig. 3 Rescheduling flowchart

During step 1 of the scheduling process, the tool creates an object containing free time slots, which include start and end timestamps. Step 2 takes care of generating an object that includes the number of pomodoros that can be scheduled in each of the free time slots along with short (5 minute) and long (20 minute) breaks. The final step queries for all the projects and tasks that user is working on, and populates them into the free time slots based on the user specified priority.

Rescheduling happens whenever users decide to skip a task, request more time for a task, or add a new task or a calendar event. Rescheduling is using a variation of the above steps, but additionally it keeps track of how many pomodoros it took for a task to complete.

If the task was skipped, all the subsequent tasks are moved up by one pomodoro and the current task is moved to the back of the line (Fig. 4). If another pomodoro was requested for the current task, all tasks are shifted down by one pomodoro and new slot is created right after the current task in progress

(Fig. 5). When a new task is added to the list, the free slot is allocated for it and pomodoro is added to the schedule. Whenever a user makes changes to the calendar by adding meetings, events, or time off, those time slots are added to the database with a *personal* tag, which lets PomaFocus skip those slots during the scheduling process.

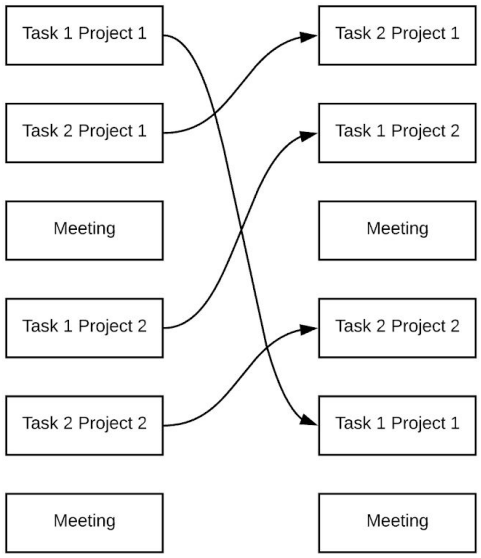

Fig. 4 Rescheduling of skipped task. Left column represents current schedule and right new schedule.

The most difficult part of scheduling is developing the rescheduling code because it deals with moving tasks around existing items. The difficulty is in keeping track of the short and long breaks, along with the necessity of shifting tasks to next/previous day. The start and end times, present in task object, help to mitigate this difficulty, because the fields are used for shifting tasks up/down the schedule without a need to recalculate the start/end times. Howerever, it's important to keep the work day preferences of the user and number of tasks in mind while updating task objects. When rescheduling occurs the tasks must be accurately shifted to the next day, if current day doesn't have free time slots. To do that, rescheduling code is verifying that task's start/end times fall into user's work day preferences. Additionally, to ensure that breaks are properly assigned, while rescheduling must keep count of tasks updated. This ensures that every task gets a short break and a long break after every four tasks.

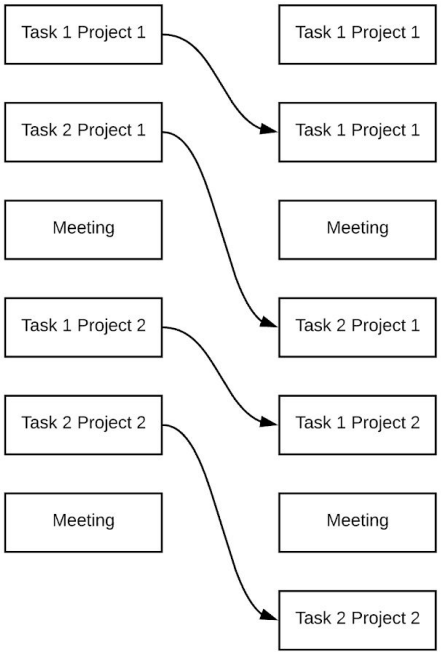

Fig. 5 Rescheduling of snoozed task. Left column represents current schedule and right new schedule.

IV. User Experience (UX) Considerations

Design of an application does not stop at the backend architecture — it is critical to also consider the look-and-feel of an application and how easy it is for users to interact with it [7]. In PomaFocus, it was important to follow Jakob Nielsen's 10 usability heuristics for User Interface [8]. According to Nielsen, it is important for a user experience to include visibility of system status, match between system and the real world, user control and freedom, consistency and standards, error prevention, recognition rather than recall, aesthetic and minimalist design, help users recognize and recover from errors, and have help and documentation. It is essential for users to have the ability to see the most important aspect of an application — in case of PomaFocus, their schedule, which is display on the home screen after logging in. Another essential aspect of a successful user experience is to ensure users are not only able to achieve a goal with a minimal number of clicks, but with each click containing a significant action.

PomaFocus uses this approach for all its features, such as creating a project or task and editing them. At any given time, a user should feel comfortable knowing where they are, what pages they came from, and what action on a specific UI element will trigger. All of this makes for an elegant UI, and provides a UX that is minimal and makes application adoption extremely simple.

V. Sample use-case flow

Below is a sample use case which lists the steps a typical user will go through:

1) Configure PomaFocus: Configuration step will collect some basic information as below, which would help PomaFocus to schedule the tasks. An Interactive bot collects the following information to configure the account:

a) Size of Pomodoro (20-30 minutes)
b) Size of small breaks (4-8 minutes)
c) Size of longer breaks (15-25 minutes)
d) When the work date start?
e) Lunch break time

2) Once PomaFocus is configured, a user will start adding projects using a template similar to the one given below:

a) User creates a project and adds task(s) to a project providing below details
   i) Priority
   ii) Completion date
   iii) Pomofocus weight

3) Upon completion of a single Pomodoro, the application sends a notification to the user. Users can either select "Finished" or "Snooze". On snooze, PomaFocus will reschedule the task by allocating one more pomodoro. Additionally, PomaFocus schedules a break interval between each task.

4) Initially every task will be scheduled for one pomodoro. But as user continues to use PomaFocus, it will adjust the pomodoros based on the learnings from previous tasks

VI. Future work

Integration of such Task management solutions with industry standard tools like Jira used for Software development methodologies would offer great value. With such integration, tasks can be pulled from existing tools used for project management and can be scheduled automatically for all task owners and will surely help with increased productivity. The reporting tools developed on top of such integration can give a clear picture of overall progress across the organization.

VII. Conclusions

There has been a lot of studies proving the success of the Pomodoro Technique. But the need of time is to have some tool which helps users to use the technique efficiently. The proposed solution solves this problem by developed scheduling algorithm and makes it more efficient, scalable and cost effective by adopting Cloud Architecture.

Acknowledgment

We would like to thank Serverless-Stack [9] forum for creating good learning material around Serverless framework.

References

[1] Cirillo, F., "The Pomodoro Technique. XPLabs Technical Report version 1.3. English Version", June 2007.

[2] Feng, Jianying, "An evaluation of the Pomodoro Technique for stopping procrastination and behaviour change.", 2016.

[3] Federico Gobbo and Matteo Vaccari, "The Pomodoro Technique for Sustainable Pace in Extreme Programming Teams", June 2008.

[4] Xiaofeng Wang, Federico Gobbo and Michael Lane, "Turning Time from Enemy into an Ally Using the Pomodoro Technique", May 2010.

[5] Ariel Ortiz, "Architecting Serverless Microservices on the Cloud with AWS", February 2019.

[6] Ioana Baldini, Paul Castro, Kerry Chang, Perry Cheng, Stephen Fink, Vatche, Ishakian, Nick Mitchell, Vinod Muthusamy, Rodric Rabbah, Aleksander Slominski and Philippe Suter, "Serverless Computing: Current Trends and Open Problems", December 2017.

[7] Benyon, David. "Designing interactive systems: A comprehensive guide to HCI, UX and interaction design", 2014.

[8] Nielsen, Jakob, "10 usability heuristics for user interface design", 1995.

[9] Serverless Stack, https://serverless-stack.com/