EFFICIENT TIME MANAGEMENT IN THE DIGITAL ERA: REINFORCEMENT LEARNING IN ACTION

A Project Report

Submitted by:

N. Nagasatya Sai (208T1A05G4)

T. Srihari (208T1A05I1)

K. Bindu Sri (208T1A05F0)

L. Manogna Swetha (208T1A05F4)

D. Lakshmi Sasi Rekha (208T1A05D8)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

at

Under the esteemed guidance of

Mrs. Gummadi Subhashini

Assistant Professor

DHANEKULA INSTITUTE OF ENGINEERING AND TECHNOLOGY

GANGURU, A.P. (INDIA) - 521139

AFFILIATED TO JNTUK, KAKINADA, ANDHRA PRADESH (INDIA)

APRIL & 2024

DECLARATION

I hereby declare that the project entitled "**EFFICIENT TIME MANAGEMENT IN THE DIGITAL ERA: REINFORCEMENT LEARNING IN ACTION**" submitted for the B. Tech. (CSE) degree is my original work and the project has not formed the basis for the award of any other degree, diploma, fellowship, or any other similar titles.

Student names	**Signature of the Student**
N. Nagasatya Sai	
T. Srihari	
K. Bindu Sri	
L. Manogna Swetha	
D. Lakshmi Sasi Rekha	

Place: Vijayawada

Date:

DHANEKULA INSTITUTE OF ENGINEERING AND TECHNOLOGY

GANGURU, A.P. (INDIA) - 521139

AFFILIATED TO JNTUK, KAKINADA, ANDHRA PRADESH (INDIA)

CERTIFICATE

This is to certify that the project titled "**EFFICIENT TIME MANAGEMENT IN THE DIGITAL ERA : REINFORCEMENT LEARNING IN ACTION**" is the bonafide work carried out by **N. Nagasatya Sai (208T1A05G4), T. Srihari (208T1A05I1), K. Bindu Sri (208T1A05F0), L. Manogna Swetha (208T1A05F4), D. Lakshmi Sasi Rekha (208T1A05D8)** are students of B Tech (CSE) of Dhanekula Institute of Engineering and Technology, affiliated to JNT University, Kakinada, AP(India) during the academic year 2020-24, in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology(Computer Science and Engineering) and that the project has not formed the basis for the award previously of any other degree, diploma, fellowship or any other similar title.

SIGNATURE OF THE GUIDE

Mrs. Gummadi Subhashini
(Asst. Professor, CSE)

SIGNATURE OF THE HOD

Dr. K. Sowmya
(HOD & Professor)

EXTERNAL EXAMINER

VISION – MISSION - PEOs

Institute Vision	Pioneering Professional Education through Quality
Institute Mission	Providing Quality Education through state-of-art infrastructure, laboratories and committed staff. Moulding Students as proficient, competent, and socially responsible engineering personnel with ingenious intellect. Involving faculty members and students in research and development works for betterment of society.
Department Vision	To empower students of Computer Science and Engineering Department to be technologically adept, innovative, global citizens possessing human values.
Department Mission	Encourage students to become self-motivated and problem-solving individuals. Prepare students for professional career with academic excellence and leadership skills. Empower the rural youth with computer education. Create Centre's of excellence in Computer Science and Engineering
Program Educational Objectives (PEOs)	Graduates of B.Tech (Computer Science & Engineering) will be able to **PEO1:** Excel in Professional career by demonstrating the capabilities of solving real time problems through Computer-based system, Machine learning and allied software applications. **PEO2:** Able to pursue higher education and research. **PEO3:** Communicate effectively, recognize, and incorporate appropriate tools and technologies in the chosen profession. **PEO4:** Adapt to technological advancements by continuous learning, team collaboration and decision making.

POs

1	**Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.
2	**Problem analysis**: Identify, formulate, review research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3	**Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4	**Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5	**Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.'
6	**The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7	**Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9	**Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11	**Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12	**Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Specific Outcome Statements (PSO`s):

1	Have expertise in algorithms, networking, web applications and software engineering for efficient design of computer-based systems of varying complexity.
2	Qualify in national international level competitive examinations for successful higher studies and employment.

PROJECT MAPPINGS

Batch No:	C4
Project Title	EFFICIENT TIME MANAGEMENT IN THE DIGITAL ERA: REINFORCEMENT LEARNING IN ACTION
Project Domain	Machine Learning
Type of the Project	Application
Guide Name	Mrs. Gummadi Subhashini
Student Roll No	Student Name
208T1A05G4	N. Nagasatya Sai
208T1A05I1	T. Srihari
208T1A05F0	K. Bindu Sri
208T1A05F4	L. Manogna Swetha
208T1A05D8	D. Lakshmi Sasi Rekha

COURSE OUTCOMES: At the end of the Course/Subject, the students will be able to

CO. No	Course Outcomes (COs)	POs	PSOs	Blooms Taxonomy & Level
R20C501.1	Identify the real-world problem with a set of requirements to design a solution.	1,2,3,6, 7, 8, 9, 10, 11	1,2	Level-3 Applying
R20C501.2	Implement, Test and Validate the solution against the requirements for a given problem.	1,2,3,4,5,6,7, 9, 10	1,2	Level-4 Analyzing
R20C501.3	Lead a team as a responsible member in developing software solutions for real world problems and societal issues with ethics.	1,3,4,8,9,10,11,12	1,2	Level-4 Analyzing
R20C501.4	Participate in discussions to bring technical and behavioral ideas for good solutions.	1,2,3,4,5,6,9,10	1,2	Level-5 Evaluating
R20C501.5	Express ideas with good communication skills during presentations.	7,9,10,11,12	1,2	Level-6 Creating
R20C501.6	Learn new technologies to contribute in the software industry for optimal solutions	3, 11, 12	1,2	Level-6 Creating

Course Outcomes vs PO`s Mapping:

Courses Out Comes	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P011	P012
R20C501.1	3	3	3	-	-	3	3	3	3	3	3	-
R20C501.2	3	3	3	3	3	3	3	-	3	3	-	-
R20C501.3	3	-	3	3	-	-	-	3	3	3	3	3
R20C501.4	3	3	3	3	3	3	-	-	3	3	-	-
R20C501.5	-	-	-	-	-	-	3	-	3	3	3	3
R20C501.6	-	-	3	-	-	-	-	-	-	-	3	3
Total	12	9	15	9	6	9	9	6	15	15	12	9
Average	3	3	3	3	3	3	3	3	3	3	3	3

Justification of Mapping of Course Outcomes with Program Outcomes:

1. R20C501.1 is strongly linked with PO1, PO2, PO3, PO6, PO7, PO8, PO9, PO10, and PO11 because we are using engineering knowledge, doing problem analysis, design and development of solutions, building the solutions which bridge gap between engineer and society, developing solutions which useful environment and promoting sustainable development, following ethics, performing the tasks both individually and teamwork, communicating for synchronizing the work, managing project and taking the requirements which are financially feasible.
2. R20C501.2 is strongly mapped with PO1, PO2, PO3, PO4, PO5, PO6, PO7, PO9 and PO10 because we use engineering concepts and research solutions for time management problem by implement, test and validate for problem. Also, we use the latest tools, follow professional ethics, communicate effectively & manage the project.
3. R20C501.3 is strongly linked with PO1, PO3, PO4, PO8, PO9, PO10, PO11, and PO12 because engineering knowledge is applied to solve time management problems while considering society and the environment. Also, we follow professional ethics, communicate in team for solutions and manage project.
4. R20C501.4 is like PO1, PO2, PO3, PO4, PO5, PO6, PO9, PO10, because we apply engineering knowledge to solve problems, select the right formulas, consider society and the environment, as well as safety and ethics. Also, each individual communicates with one another for good solutions and manages project.

5. R20C501.5 is strongly connected to PO7, PO9, PO10, PO11 and PO12 because use solution created is helpful for the individual in the environment, follow professional rules, work alone or in a team, communicate to solve problems, and understand the importance of learning newer technological change.
6. R20C501.6 is strongly mapped with PO3, PO11, and PO12 because we are designing and developing the solution and learn new technologies to contribute in the software industry for optimal solutions. Also, we understand importance of learning newer technological change.

Course Outcomes vs PSOs Mapping:

Courses Outcomes	PSO1	PSO2
R20C501.1	3	3
R20C501.2	3	3
R20C501.3	3	3
R20C501.4	3	3
R20C501.5	3	3
R20C501.6	3	3
Total	18	18
Average	3	3

Justification of Mapping of Course Outcomes with Program Specific Outcomes:

All CO's are strongly mapped with PSO1 and PSO2 because we have done this project with having expertise in algorithms, networking, web applications and software engineering for efficient design and it is also helpful in national and international level competitive examinations for successful higher studies and employment.

Mapping Level	Mapping Description
1	Low Level Mapping with PO & PSO
2	Moderate Mapping with PO & PSO
3	High Level Mapping with PO & PSO

ACKNOWLEDGEMENT

Behind every achievement lies an unfathomable sea of gratitude to those who activated it, without whom it would ever have come into existence. To them we lay the words of gratitude imprinted with us.

We would like to thank our respected Principal, **Dr. RAVI KADIYALA** and, **Dr. K. SOWMYA**, Head of the Department, Computer Science and Engineering for their support throughout our major project.

It is our sincere obligation to thank our guide, **Mrs. GUMMADI SUBHASHINI**, Department of Computer Science and Engineering, for her timely valuable guidance and suggestions for this major project.

We would like to express our immense pleasure in expressing an immeasurable sense of gratitude to **Mr. M. RAVIKANTH**, Assistant Professor and Project Coordinators for giving opportunity to make this project a successful one

We also extend our thanks to all the faculty members of the Computer Science & Engineering department for their valuable contributions in this project.

We would like to extend our warm appreciation to all our friends for sharing their knowledge and valuable contributions in this project.

Finally, we express our deep sense of gratitude to our parents for their continuous support throughout our academic career and their encouragement in the completion of this project successfully.

N. Naga Satya Sai	(208T1A05G4)
T. Srihari	(208T1A05I1)
K. Bindu Sri	(208T1A05F0)
L. Manogna Swetha	(208T1A05F4)
D. Lakshmi Sasi Rekha	(208T1A05D8)

ABSTRACT

In the contemporary digital landscape, effective time management is paramount for individuals across various domains. This project explores a novel approach to tackling time management problems through Actor-Critic Reinforcement learning. Reinforcement learning, a powerful machine learning paradigm, is the core methodology. It serves as the foundation for developing an intelligent time management system that adapts to the unique behaviors and preferences of individuals. This system provides personalized time management strategies, enhancing productivity and overall effectiveness. Unlike conventional methods, which may involve static scheduling models, this project embraces dynamic learning and adaptation through reinforcement learning algorithms. By doing so, it offers a flexible solution for the students to optimally manage their time via efficient planning of tasks. In summary, the project seeks to address the universal issue of efficient time management in the digital age by harnessing the capabilities of reinforcement learning. By introducing this innovative approach, we aim to equip individuals from diverse fields with tools to thrive in an increasingly digitized world.

Keywords: Time management, Reinforcement learning, Adaptive Scheduling, through Actor-Critic Reinforcement learning, Create Optimal Schedule.

List of Figures

List of Tables

CONTENTS

INTRODUCTION

INTRODUCTION

In the fast-evolving digital landscape of today, the ability to manage time efficiently stands as a cornerstone skill for individuals navigating diverse professional and personal spheres. This Project undertakes a comprehensive exploration of an innovative approach aimed at confronting the multifaceted challenges of time management, employing Actor-Critic Reinforcement Learning as its central framework. Reinforcement learning, a sophisticated machine learning paradigm renowned for its adaptability and robustness, serves as the linchpin of this endeavour, offering a powerful methodology to construct an intelligent time management system tailored to individual needs. At the heart of this system lies the concept of personalized time management strategies, meticulously crafted to accommodate the nuanced behaviours and preferences of each user. Unlike traditional approaches characterized by rigid, one-size-fits-all scheduling models, the methodology advocated in this project prioritizes dynamic learning and adaptation facilitated by reinforcement learning algorithms. By leveraging this dynamic framework, the system endeavours to furnish users, particularly students, with the tools necessary to optimize their time allocation and task prioritization, thereby fostering enhanced productivity and efficacy in their endeavours. A departure from conventional wisdom, which often espouses static planning paradigms, this project champions a paradigm shift towards agility and responsiveness in time management practices. Through its innovative utilization of reinforcement learning, the proposed system offers a malleable and adaptable solution capable of accommodating the fluid nature of modern-day commitments and obligations. In essence, this research initiative represents a concerted effort to address the pervasive challenge of effective time management in the digital era by harnessing the transformative potential of advanced machine learning techniques.

1.1. PROBLEM STATEMENT

The challenge is ineffective time management amid evolving commitments, causing reduced productivity, especially among students. Traditional methods lack adaptability and customization, leading to suboptimal time use. Addressing this requires innovative solutions, utilizing advanced machine learning like Actor-Critic Reinforcement Learning, to create personalized systems capable of dynamically adjusting to changing circumstances, improving productivity without resorting to plagiarism.

1.2. PROJECT OVERVIEW

Personalized time management: At the core of this system are personalized time management strategies meticulously designed to accommodate the nuanced behaviours and preferences of each user. Unlike traditional rigid scheduling models, this approach prioritizes dynamic learning and adaptation facilitated by reinforcement learning algorithms.

Reinforcement Learning: Reinforcement learning, a sophisticated machine learning paradigm known for its adaptability and robustness, serves as the central methodology in constructing an intelligent time management system tailored to individual needs.

Dynamic learning Adaptation: Unlike static planning paradigms, which remain fixed regardless of changes in circumstances, personalized time management strategies prioritize dynamic learning and adaptation. Leveraging reinforcement learning algorithms, the system continuously refines its recommendations based on user feedback and evolving priorities.

Enhancing Productivity and efficacy: By leveraging this dynamic framework, the system aims to equip users, particularly students, with the tools necessary to optimize time allocation and task prioritization. This, in turn, fosters enhanced productivity and efficacy in their endeavours.

Reinforcement learning: Reinforcement learning (RL) is a type of machine learning paradigm where an agent learns to make decisions by interacting with an environment in order to maximize some notion of cumulative reward. In the context of time

management, reinforcement learning can be applied to develop intelligent systems that dynamically adapt and optimize time allocation and task prioritization based on user feedback and environmental changes.

1. **Agent:** The individual seeking to manage their time effectively is represented as the RL agent. The agent's goal is to make decisions (such as allocating time to different tasks) in a way that maximizes long-term rewards (such as productivity or satisfaction).
2. **Environment:** The environment in this case includes all the tasks, commitments, and activities that the individual needs to manage within their schedule. The environment provides feedback to the agent based on its actions, which helps the agent learn and improve its decision-making process.
3. **Actions:** Actions in RL represent the choices that the agent can make at each time step. In the context of time management, these actions might include allocating time to specific tasks, scheduling events, or adjusting priorities.
4. **Rewards:** Rewards are used to provide feedback to the agent about the quality of its actions. In time management, rewards could be based on the completion of tasks, adherence to deadlines, or achieving personal goals. The agent learns to maximize cumulative rewards over time by adjusting its behaviour accordingly.
5. **Learning Process:** Through trial and error, the RL agent learns which actions lead to favourable outcomes and which ones do not. By experiencing the consequences of its actions and receiving feedback in the form of rewards, the agent gradually improves its decision-making process and becomes more adept at managing time effectively.
6. **Policy:** The policy in RL represents the agent's strategy for choosing actions based on its current state. Over time, the agent learns an optimal policy that dictates the best course of action in different situations, ultimately leading to improved time management skills.

Actor-Critic algorithm: The Actor-Critic algorithm is a reinforcement learning technique that combines elements of both value-based and policy-based methods. In the context of time management, the Actor-Critic algorithm can be utilized to develop an intelligent system that learns to make decisions regarding task prioritization and time allocation.

1. **Actor:** The "Actor" in the Actor-Critic algorithm is responsible for learning a policy, which is a mapping from states (current situations) to actions (decisions). In the context of time management, the Actor learns to select optimal actions (such as allocating time to different tasks) based on the current state of the individual's schedule and goals.
2. **Critic:** The "Critic" in the Actor-Critic algorithm evaluates the actions chosen by the Actor by estimating the expected cumulative rewards (or value) associated with those

   actions. In time management, the Critic assesses the quality of the decisions made by the Actor by considering factors such as task completion, adherence to deadlines, and overall productivity.
3. **Policy Gradient :** The Actor learns to improve its policy by following the gradient of the expected cumulative rewards with respect to the policy parameters. In other words, the Actor adjusts its decision-making strategy based on the feedback received from the Critic, with the goal of maximizing long-term rewards.
4. **Temporal Difference Learning:** The Critic updates its value estimates based on the difference between the expected cumulative rewards predicted by the current value function and the rewards actually received. This allows the Critic to learn from experience and refine its estimates of the value of different actions in various situations.
5. **Actor-Critic Interaction:** The Actor and Critic interact iteratively, with the Actor selecting actions based on its policy and the Critic providing feedback on the quality of those actions. Through this iterative process, both components of the algorithm learn to improve their respective functions, ultimately leading to better decision-making in time management.

1.3 Hardware Specification:

1. Compute/Server: A Standard laptop or desktop for development and experimentation.
2. Processor (CPU): A multi-core processor (e.g., Intel Core i5/i7 or AMD Ryzen) for faster training and experimentation.
3. Memory (RAM): At least 8GB of RAM is recommended.
4. Storage: Adequate storage space (e.g., 256GB SSD or larger) for storing datasets, model checkpoints, and code.
5. Internet Connectivity: A stable internet connection for accessing online resources, downloading libraries, and collaborating with others.
6. Graphics Processing Unit (GPU)

1.4 Software Specification:

1. Operating System: Windows 10/11 for development and Linux for deployment.
2. Programming language: Python. 3. Development Environment : Visual Studio code, Jupyter notebook.
3. Machine learning libraries: Pytorch.
4. Reinforcement learning frameworks: OpenAI Gym, Stable Baselines.
5. Data Management and Analysis: Pandas, Matplotlib.
6. Version Control: Git and GitHub.
7. Frontend: HTML, CSS, JavaScript, Bootstrap.
8. Backend: Django

1.5 Basic Concepts

Task Scheduling:

Task scheduling undergoes a transformative evolution through the integration of reinforcement learning. This innovative approach harnesses the power of machine learning algorithms to optimize scheduling decisions, enhancing productivity and resource utilization. At its core, the process involves the utilization of historical data and user feedback to train a reinforcement learning model. This model learns from past scheduling patterns, considering various factors such as task duration, deadlines, priorities, and dependencies. Through iterative learning and optimization, the algorithm refines its scheduling policies, adapting to dynamic environments and evolving user preferences.

The essence of task scheduling in this project lies in its ability to balance competing objectives, such as meeting deadlines while maximizing efficiency. By continuously evaluating and adjusting scheduling decisions based on real-time feedback, the system becomes adept at allocating resources effectively and minimizing time wastage. Ultimately, the goal is to create a seamless workflow where tasks are scheduled intelligently, allowing individuals and organizations to achieve their objectives with greater ease and effectiveness in the fast-paced digital landscape.

Task Rescheduling:

Task rescheduling operates as a continual process of adaptation and optimization, driven by intelligent algorithms trained through reinforcement learning. When disruptions or changes occur, such as new priorities, unexpected delays, or resource constraints, the system dynamically reevaluates its scheduling decisions to maintain efficiency and meet objectives.

The task rescheduling mechanism functions by first assessing the impact of the change on existing schedules and priorities. It considers factors such as task dependencies, deadlines, and resource availability to generate new schedules that align with updated objectives.

Once a new schedule is proposed, the system evaluates its potential outcomes through simulation or probabilistic modeling, assessing its feasibility and impact on overall performance. Feedback mechanisms allow the system to learn from past rescheduling decisions, refining its strategies over time to become more adept at handling unforeseen events. Through this iterative process of adaptation and learning, the project empowers individuals and organizations to navigate the complexities of the digital landscape with agility and efficiency, ensuring that time is optimally utilized even amidst constant change.

Creating a Schedule

Creating the schedule includes following actions

- List all tasks: This includes school assignments, extracurricular activities, personal commitments, and even leisure time.
- Estimate task duration: Be realistic about the time required for each task, considering complexity and research needs.
- Prioritize tasks: Identify urgent and important tasks, scheduling them first. Consider factors like deadlines, difficulty, and learning benefits.
- Block time slots: Allocate specific time slots in a planner or calendar app for each task, factoring in breaks and unforeseen events.

Benefits of Task Scheduling:

- Increased productivity: A clear schedule eliminates procrastination and allows students to focus on specific tasks during dedicated time blocks.
- Reduced stress: Feeling overwhelmed by a to-do list can be paralyzing. Scheduling helps students feel in control and prioritize tasks, mitigating stress.
- Improved academic performance: By dedicating time for studying and completing assignments, students are more likely to meet deadlines and retain information.

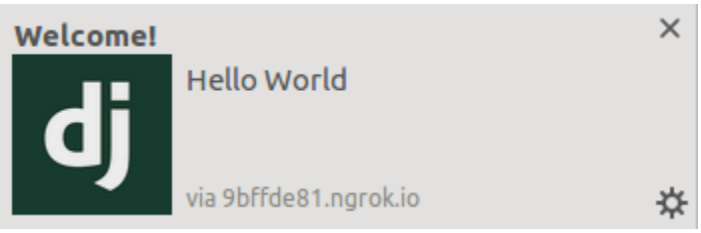
Notifications:

Notifications play a crucial role in keeping users informed and reminding them of important tasks or events. The notification system in this project is designed to be intelligent and personalized. It considers the user's preferences, priorities, and schedule. By analyzing data and patterns, the system determines the most appropriate time to send notifications, ensuring that they are not intrusive or overwhelming.

The system utilizes various factors to determine the timing of notifications. These factors include the urgency of the task, the user's availability, and the context in which the task needs to be completed. For example, if a task has a deadline approaching, the system may send a notification in advance to remind the user to complete it. Additionally, the notification system can adapt and learn from user behavior. It can observe how users interact with notifications and adjust its timing and frequency accordingly. This adaptive approach ensures that users receive notifications when they are most likely to engage with them. Overall, to provide users with timely and relevant reminders, helping them manage their time efficiently.

Django Webpush

- Django-Webpush is a Package made for integrating and sending Web Push Notification in Django Application.
- Currently, it Supports Sending Push Notification to **Firefox 46+, Chrome 52+ and Apple devices on iOS 16.4+**.
- django-webpush is shipped with built in **jinja** support. If you would like to use with jinja backend, pass pipeline.jinja2.PipelineExtension to your jinja environment.
- A Web Push generally have a header and body. According to the W3C Specification, the data should be encrypted in transmission. The data is addressed as payload generally. Also, a TTL header should be included indicating how much time the web push server store the data if the user is not online.



Celery

Celery is an open-source, asynchronous task queue or job queue. It's written in Python and is cross-platform. Celery is used in production systems, such as Instagram, to process millions of tasks every day. Celery's features include:

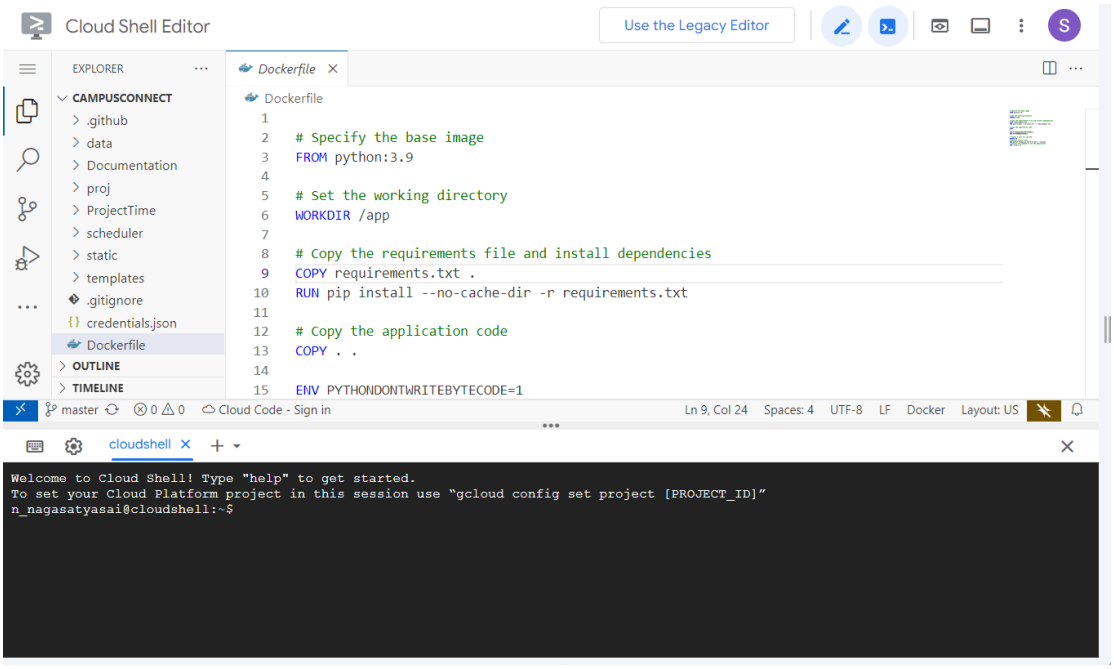
- Asynchronous task queue
- Distributed message passing
- Scheduling
- Real-time operations
- Multiprocessing, eventlet, or gevent
- Asynchronous and synchronous execution

Celery is a task queue with batteries included. It’s easy to use so that you can get started without learning the full complexities of the problem it solves. It’s designed around best practices so that your product can scale and integrate with other languages, and it comes with the tools and support you need to run such a system in production. Celery is on the Python Package Index (PyPI), so it can be installed with standard Python tools like pip install celery.

Google Cloud Shell

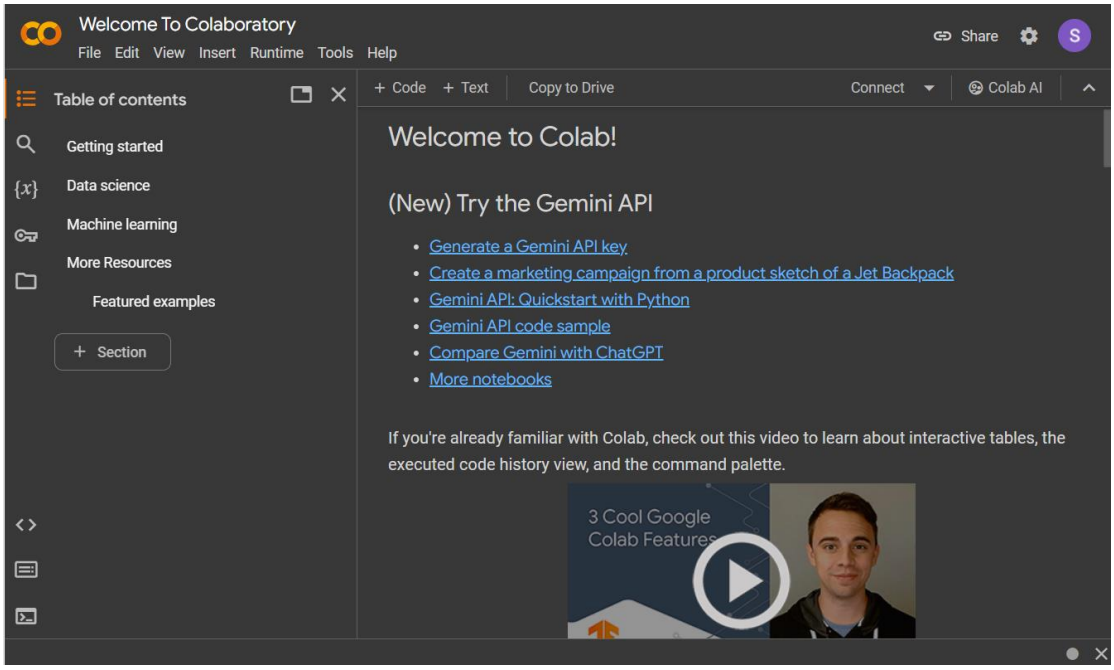
- **Google Cloud Shell** is an online bash shell based on Debian. The free tier (included with all Gmail accounts) includes 8 gigabytes of random-access memory and a persistent 5 gigabyte home directory. Except for the home and root directories, the Cloud Shell environment is volatile.

- Google Cloud Shell supports OpenSSH for secure remote access and integrates seamlessly with the Google Cloud Command-Line Interface (CLI) for managing cloud resources. Additionally, it features a code editor based on Eclipse Theia, making it a versatile tool for developers and system administrators.



Google Colab

- **Project Jupyter** is a project to develop open-source software, open standards, and services for interactive computing across multiple programming languages.
- Colab is a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education.
- It was spun off from I Python in 2014 by Fernando Pérez and Brian Granger. Project Jupyter's name is a reference to the three core programming languages supported by Jupyter, which are Julia, Python and R. Its name and logo are an homage to Galileo's discovery of the moons of Jupiter, as documented in notebooks attributed to Galileo.



Applications

- Jupyter Notebook (formerly IPython Notebook) is a web-based interactive computational environment for creating notebook documents. Jupyter Notebook is built using several open-source libraries, including IPython, ZeroMQ, Tornado, jQuery, Bootstrap, and MathJax. A Jupyter Notebook application is a browser-based REPL containing an ordered list of input/output cells which can contain code, text (using Github Flavored Markdown), mathematics, plots and rich media.
- Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica, and Sage Math, a computational interface style that originated with Mathematica in the 1980s. Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.
- Jupyter Lab is a newer user interface for Project Jupyter, offering a flexible user interface and more features than the classic notebook UI. The first stable release was announced on February 20, 2018. In 2015, a joint $6 million grant from The Leona M. and Harry B. Helmsley Charitable Trust, The Gordon and Betty Moore Foundation, and The Alfred P. Sloan Foundation funded work that led to expanded capabilities of the core Jupyter tools, as well as to the creation of Jupyter Lab.

- GitHub announced in November 2022 that Jupyter Lab would be available in its online Coding platform called Code space.
- In August 2023, Jupyter AI, a Jupyter extension, was released. This extension incorporates generative artificial intelligence into Jupyter notebooks, enabling users to explain and generate code, rectify errors, summarize content, inquire about their local files, and generate complete notebooks based on natural language prompts.
- Jupyter Hub is a multi-user server for Jupyter Notebooks. It is designed to support many users by spawning, managing, and proxying many singular Jupyter Notebook servers.

Google Calendar API

- **Google APIs** are application programming interfaces developed by Google which allow communication with Google Services and their integration to other services. Examples of these include Search, Gmail, Translate or Google Maps. Third-party apps can use these APIs to take advantage of or extend the functionality of the existing services.
- The APIs provide functionality like analytics, machine learning as a service (the Prediction API) or access to user data (when permission to read the data is given). Another important example is an embedded Google map on a website, which can be achieved using the Static Maps API, Places API or Google Earth API.

The Google Calendar API empowers developers to programmatically manage calendars and integrate scheduling functionalities into their applications. This translates to automating tasks, streamlining workflows, and boosting overall efficiency.

Resource Types:

- **Calendars:** Represent individual calendars (work, personal, team, etc.). The API allows you to list, retrieve, and even create new calendars.

- **Events:** The heart of calendar management. Events encompass details like title, description, date, time, attendees, and recurrence. You can create, read, update, and delete events using the API.
- **Attendees:** Represent participants in an event. The API facilitates managing attendees by sending invites, tracking RSVPs, and updating attendance status.
- **ACL (Access Control Lists):** Define permissions for users to access and modify specific calendars. The API allows managing ACLs for granular control.

Key Functionalities:

- **CRUD Operations for Events:** Create, Read (get details), Update (modify details), and Delete events.
- **Event Search:** Find specific events based on various criteria like time, date, title, attendees, or keywords in the description.
- **Calendar List Retrieval:** Get a list of all calendars associated with the authorized Google account.
- **Attendee Management:** Invite participants, track their responses, and update their status (accepted, declined, maybe).
- **Free/Busy Information:** Check availability of attendees for scheduling conflicts.
- **Drive Integration:** Attach relevant Google Drive files (Docs, Sheets, Slides) to events for easy access by attendees.

Benefits:

- **Automation:** Automate repetitive scheduling tasks like sending meeting invites or booking appointments. This frees up developers and users for more strategic work.
- **Customization:** Integrate calendar functionalities seamlessly into your applications, tailoring the experience to your specific needs.
- **Improved Workflow:** Streamline scheduling processes by offering features like centralized calendar management, automated reminders, and conflict detection.

- **Enhanced Collaboration:** Facilitate better communication and coordination by allowing easy scheduling of meetings and sharing relevant event details with attendees.
- **Accessibility:** Manage calendars and events from anywhere with an internet connection, promoting remote work and team collaboration.

Getting Started:

1. **Enable the Google Calendar API:** Activate the API in the Google Cloud Console for your project.
2. **Create Credentials:** Generate OAuth credentials to authorize your application to access Google Calendar data on behalf of a user.
3. **Choose a Client Library:** Google provides client libraries in various programming languages (Python, Java, JavaScript etc.) for easier interaction with the API.
4. **Make API Requests:** Use the chosen client library to make HTTP requests to the Calendar API endpoints for desired functionalities (creating events, retrieving calendar list etc.).

Security Considerations:

- **OAuth Consent:** OAuth ensures secure access by prompting users to grant permission to your application before accessing their calendar data.
- **Scopes:** API scopes define the level of access your application requests. Choose scopes that are strictly necessary for your application's functionality to minimize potential security risks.

Django

Django is a free and open-source web framework built with Python, renowned for its rapid development, clean design, and robust security features [11]. Let's delve into its core functionalities and explore why it's a favourite among developers.

Why Django?

- **Faster Development:**
    - **Pre-built components:** Django offers a treasure trove of built-in features like user authentication, forms, content management, and more, reducing development time.
    - **DRY (Don't Repeat Yourself) principle:** By eliminating repetitive code, Django promotes clean and maintainable codebases.
- **Security:** Django prioritizes security by offering features like automatic escaping to prevent common web vulnerabilities like Cross-Site Scripting (XSS).
- **Scalability:** Django is built to handle even the most demanding web applications, making it suitable for small-scale projects to large-scale enterprise applications.
- **Large and Active Community:** Benefit from extensive documentation, tutorials, and a vast community of developers for support and collaboration.

Architectural Overview

- **Model-View-Template (MVT) Pattern:** Django adheres to the MVT pattern, providing a clear separation of concerns:
    - **Models:** Define the data structure of your application (users, products, blog posts etc.).
    - **Views:** Handle user requests and interact with models to retrieve or manipulate data.
    - **Templates:** Contain the presentation logic and dictate how data is displayed to the user.

- **URL Routing:** Django's URL routing system maps URLs to specific views in your application, ensuring requests reach the appropriate handler.

Key Features of Django:

- **Admin Interface:** A built-in admin interface allows for easy management of data models through a user-friendly web interface.
- **Object-Relational Mapper (ORM):** The ORM simplifies interaction with databases by providing an elegant way to work with data models in Python code.
- **Form Handling:** Django offers robust form handling functionalities for user input validation and data protection.
- **Template Engine:** The powerful templating engine provides a flexible way to generate HTML markup with dynamic content from your models.
- **Middleware:** Middleware components intercept requests and responses, enabling functionalities like user authentication, session management, and security checks.

Django Recurrence

Django, a popular web framework, offers built-in functionality for managing recurring events through the django-recurrence package [12]. This presentation dives deep into how you can leverage this package to create powerful scheduling features within your Django applications.

Why Recurring Events?

- **Reduced Code:** Eliminate the need to manually write logic for repetitive scheduling tasks.
- **Flexibility:** Define various recurrence patterns (daily, weekly, monthly, yearly) with custom intervals.
- **Improved User Experience:** Allow users to easily set up recurring events like meetings, appointments, or reminders.

Key Concepts:

- **RecurrenceRule:** The core component, defining the scheduling pattern. It specifies:
    - **Frequency:** How often the event repeats (daily, weekly, etc.)
    - **Interval:** The number of times the frequency unit elapses between occurrences (e.g., every other week).
    - **Weekday (optional):** For weekly occurrences, specify the specific day(s) (e.g., Tuesday, Thursday).
    - **Start and End Dates (optional):** Define the timeframe for the recurrence (e.g., repeat for 6 months).
- **AbstractRecurringEvent:** A base class representing a recurring event. It doesn't have a database field itself but provides methods for generating occurrences.

Implementation Steps:

1. **Install django-recurren**ce: Add "recurrence" to your INSTALLED_APPS in the settings.py file.
2. **Define a Model**: Create a model for your event data (e.g., Meeting, Appointment).
3. **Add a Recurrence Field**: Include a recurrence field of type recurrence.fields.RecurrenceField to your model.
4. **Generate Occurrences**: Use the occurrences method of the recurrence field to generate a queryset of concrete event instances based on the defined recurrence rule.

HTML:

- Structure and Content: HTML (Hypertext Markup Language) is the foundation of web development, used to structure and define the content of web pages. It consists of a set of markup tags that describe the elements within a document, such as headings, paragraphs, images, links, and more.

- Semantic Markup: HTML5 introduced semantic elements that provide meaning to the content, improving accessibility and search engine optimization (SEO). Semantic tags like <header>, <nav>, <article>, and <footer> help browsers and search engines understand the structure of the page better, leading to better user experiences and improved ranking in search results.

- Cross-Browser Compatibility: HTML is supported by all modern web browsers, ensuring consistent rendering of web pages across different platforms and devices. This cross-browser compatibility is essential for reaching a broad audience and delivering a consistent user experience.

- Integration with CSS and JavaScript: HTML works seamlessly with Cascading Style Sheets (CSS) and JavaScript to enhance the presentation and interactivity of web pages. CSS is used to style HTML elements, while JavaScript adds dynamic behavior and functionality, allowing developers to create rich, interactive web experiences.

- Evolution and Standards: HTML has evolved over time, with new versions introducing new features, improvements, and best practices. The latest version, HTML5, introduced significant enhancements, including multimedia support, semantic elements, offline web applications, and more. Adhering to HTML standards ensures compatibility, accessibility, and future-proofing of web projects.

CSS:

- Styling Web Pages: CSS is a stylesheet language used to style the visual presentation of HTML elements on web pages. It enables developers to control aspects such as colors, fonts, layouts, and animations, enhancing the overall look and feel of websites.
- Cascading Nature: CSS follows a cascading style hierarchy, where styles can be applied to elements in multiple ways, such as inline styles, internal stylesheets, and external stylesheets. Styles cascade from more specific selectors to more general ones, allowing for easy customization and overriding of styles.

- Selectors and Declarations: CSS uses selectors to target HTML elements and declarations to define the styling rules for those elements. Selectors can target elements based on their tag name, class, ID, attributes, or relationship with other elements. Declarations consist of a property and a value, specifying how the targeted elements should be styled.

- Responsive Design: CSS enables the creation of responsive web designs that adapt to different screen sizes and devices. Media queries allow developers to apply different styles based on factors like screen width, height, orientation, and resolution, ensuring optimal viewing experiences across various devices.

- Modularity and Maintainability: CSS promotes modular and maintainable code by separating styles from content. By keeping styles separate, developers can easily update, reuse, and organize styling rules, leading to cleaner codebases and more efficient development workflows. Techniques like CSS preprocessors (e.g., Sass, LESS) and CSS methodologies (e.g., BEM, SMACSS) further enhance modularity and maintainability.

JavaScript:

- Client-Side Scripting Language: JavaScript is a versatile scripting language primarily used for client-side web development. It runs in web browsers and allows developers to add interactivity, dynamic behavior, and functionality to web pages. With JavaScript, developers can manipulate the DOM (Document Object Model), handle events, validate forms, create animations, and more, enhancing the user experience of web applications.

- Object-Oriented and Prototypal Language: JavaScript is an object-oriented language with support for prototypal inheritance. Objects in JavaScript can inherit properties and methods from other objects, allowing for code reuse and modularity. This flexible approach to object-oriented programming enables developers to create complex applications with ease.

- Asynchronous Programming: JavaScript is inherently asynchronous, meaning it can execute multiple operations simultaneously without blocking the main thread. Asynchronous programming in JavaScript is commonly achieved using callbacks, promises, and async/await syntax. Asynchronous operations are essential for tasks like fetching data from servers, handling user input, and performing time-consuming operations without freezing the UI.

- Cross-Platform Compatibility: JavaScript is supported by all modern web browsers, making it a cross-platform language for web development. In addition to client-side scripting, JavaScript can also be used on the server-side with platforms like Node.js. This versatility allows developers to use JavaScript for both front-end and back-end development, creating full-stack web applications using a single programming language.

BOOTSTRAP:

- Front-End Framework: Bootstrap is a popular open-source front-end framework developed by Twitter. It provides a collection of pre-designed HTML, CSS, and JavaScript components, such as buttons, forms, navigation bars, modals, and more, to help developers build responsive and visually appealing web interfaces quickly.

- Responsive Design: Bootstrap is built with responsive design principles in mind, ensuring that websites and web applications look and function well on various devices and screen sizes, including desktops, tablets, and smartphones. By using Bootstrap's grid system, developers can create flexible layouts that automatically adjust to different viewport sizes, improving user experience across devices.

- Customization and Theming: Bootstrap offers extensive customization options through its SASS (Syntactically Awesome Style Sheets) variables and mixins. Developers can easily customize Bootstrap's default styles, colors, fonts, and components to match their project's branding and design requirements. Additionally, Bootstrap provides a theming system that allows developers to create custom themes or use pre-built themes to apply consistent styling across their projects.

- Community and Ecosystem: Bootstrap has a large and active community of developers, designers, and contributors who regularly contribute to its development and maintenance. This vibrant ecosystem provides extensive documentation, tutorials, plugins, and third-party resources to support developers in using Bootstrap effectively. Additionally, Bootstrap's popularity ensures that there are plenty of templates, themes, and resources available for building websites and web applications rapidly.

PostgreSQL:

- Open-Source Relational Database Management System (RDBMS): PostgreSQL is a powerful, open-source RDBMS known for its reliability, robustness, and feature-rich capabilities. It adheres to SQL standards and provides ACID (Atomicity, Consistency, Isolation, Durability) compliance, making it suitable for handling mission-critical data in various industries.

- Advanced Features and Extensibility: PostgreSQL offers a wide range of advanced features, including support for complex data types (such as arrays, JSON, XML, and geometric data types), full-text search capabilities, geospatial functions, and advanced indexing options. Additionally, PostgreSQL's extensibility allows developers to create custom data types, functions, and procedural languages to address specific requirements.

- Scalability and Performance: PostgreSQL is designed to handle high-volume workloads and large datasets efficiently. It supports multi-version concurrency control (MVCC) for managing concurrent transactions, parallel query processing for improved performance, and advanced optimization techniques to enhance query execution speed. Furthermore, PostgreSQL can be horizontally scaled using built-in features like table partitioning and replication, as well as third-party tools like pgpool-II and Citus.

- Community and Support: PostgreSQL has a vibrant and active community of developers, users, and contributors who collaborate to improve the software, provide support, and share knowledge. The PostgreSQL Global Development Group oversees the development and maintenance of PostgreSQL, ensuring regular updates, bug fixes, and security patches. Additionally, PostgreSQL's extensive documentation, mailing lists, forums, and community events offer valuable resources for users seeking assistance or guidance.

Pytorch:

- Deep Learning Framework: PyTorch is an open-source deep learning framework primarily developed by Facebook's AI Research lab (FAIR). It provides a flexible and dynamic computational graph system, making it suitable for building and training neural networks for various machine learning tasks, including image classification, natural language processing, and reinforcement learning.

- Dynamic Computation Graphs: PyTorch adopts a dynamic computation graph approach, allowing developers to define and modify neural network architectures on-the-fly during runtime. This dynamic nature of PyTorch facilitates easier debugging, experimentation, and model customization compared to static graph frameworks like TensorFlow. Additionally, PyTorch's dynamic graph execution enables support for dynamic control flow, imperative programming style, and seamless integration with Python's ecosystem.

- Tensor Manipulation and GPU Acceleration: PyTorch provides efficient tensor operations and GPU acceleration to expedite the training and inference processes for deep learning models. Its tensor library offers a rich set of mathematical functions and operations for manipulating multi-dimensional arrays (tensors), making it convenient for performing computations on large-scale datasets. Moreover, PyTorch seamlessly integrates with CUDA, allowing developers to leverage GPU resources for parallelizing computations and accelerating model training.

- Ecosystem and Community: PyTorch has a thriving ecosystem and a supportive community of researchers, developers, and practitioners who contribute to its growth and development. The PyTorch ecosystem includes various libraries, tools, and frameworks built on top of PyTorch, such as torchvision for computer vision tasks, torchtext for natural language processing, and PyTorch Lightning for high-level abstractions and simplifying the training process.

OpenAI gym:

- Reinforcement Learning Environment: OpenAI Gym is a toolkit for developing and comparing reinforcement learning algorithms. It provides a wide range of environments, such as classic control problems, Atari games, robotics simulations, and more, allowing researchers and developers to test and benchmark their reinforcement learning algorithms in diverse settings.

- Unified API: OpenAI Gym offers a unified API (Application Programming Interface) for interacting with different environments, making it easy to train and evaluate reinforcement learning agents across various tasks. The API provides methods for interacting with environments, such as taking actions, receiving observations, and obtaining rewards, enabling seamless integration with reinforcement learning algorithms and frameworks.

- Extensible and Customizable: OpenAI Gym is highly extensible and customizable, allowing users to create their custom environments tailored to specific tasks or research domains. The Gym library provides tools and utilities for defining new environments, specifying observation spaces, action spaces, rewards, and termination conditions, empowering researchers to explore novel scenarios and challenges in reinforcement learning.

- Community and Resources: OpenAI Gym has a vibrant community of researchers, practitioners, and enthusiasts who contribute to its development, share their work, and provide support to fellow users. The Gym community hosts discussions, tutorials, code examples, and research papers, facilitating knowledge sharing and collaboration in the field of reinforcement learning. Additionally, OpenAI Gym integrates with other libraries and frameworks, such as TensorFlow, PyTorch, and RLlib, enabling users to leverage existing tools and resources for buildings and training reinforcement learning models.

Stablebaselines:

- High-Performance Reinforcement Learning (RL) Library: Stable Baselines is an open-source library built on top of OpenAI Baselines, designed to provide reliable implementations of reinforcement learning algorithms. It offers a collection of state-of-the-art RL algorithms, including deep reinforcement learning methods such as Deep Q-Networks (DQN), Proximal Policy Optimization (PPO), and Trust Region Policy Optimization (TRPO).

- Ease of Use and Integration: Stable Baselines is designed with simplicity and usability in mind, providing a user-friendly API that facilitates rapid experimentation and development of RL models. It offers clear documentation, code examples, and tutorials to help users get started quickly. Additionally, Stable Baselines seamlessly integrates with popular deep learning frameworks such as TensorFlow and PyTorch, enabling users to leverage their existing knowledge and tools for training and deploying RL models.

- Efficiency and Scalability: Stable Baselines is optimized for performance and scalability, allowing users to train RL models efficiently on single CPUs, multiple CPUs, or GPUs. It leverages parallelization techniques, vectorized operations, and distributed computing to accelerate training and handle large-scale environments and datasets effectively. Moreover, Stable Baselines offers memory-efficient implementations of RL algorithms, enabling users to train models with limited computational resources.

- Versatility and Extensibility: Stable Baselines supports a wide range of reinforcement learning environments, including classic control tasks, Atari games, robotics simulations, and custom environments created using OpenAI Gym or other frameworks. It provides flexibility for users to customize and extend existing algorithms or develop new algorithms tailored to specific tasks or research objectives. Additionally, Stable Baselines integrates with other libraries and tools in the RL ecosystem, enabling users to combine it with other components for building comprehensive RL systems.

Pandas:

- Data Manipulation and Analysis: Pandas is a powerful Python library for data manipulation and analysis. It provides data structures like DataFrame and Series, which enable users to efficiently handle and analyze tabular data. With Pandas, users can perform various data operations, including data cleaning, transformation, aggregation, and visualization.

- Integration with Data Sources: Pandas supports seamless integration with various data sources and formats, such as CSV files, Excel spreadsheets, SQL databases, JSON files, and more. It offers functions and methods for reading and writing data from/to different sources, allowing users to work with data from diverse sources within the same environment.

- Data Wrangling and Preparation: Pandas simplifies the process of data wrangling and preparation by providing a rich set of functions and methods for handling missing values, reshaping data, merging/joining datasets, grouping data, and applying transformations. These capabilities make it easier for users to prepare data for analysis and modeling tasks.

- Time Series Analysis: Pandas excels at time series analysis and manipulation, offering specialized functionalities for handling time-stamped data. Users can easily resample, slice, aggregate, and perform calculations on time series data using Pandas' time series-specific methods. Additionally, Pandas integrates seamlessly with other libraries like NumPy and Matplotlib for advanced time series visualization and analysis.

Matplotlib:

- Data Visualization: Matplotlib is a comprehensive Python library for creating static, interactive, and publication-quality visualizations. It provides a wide range of plotting functions and customization options for creating various types of plots, including line plots, scatter plots, bar plots, histograms, heatmaps, and more.

- Customization and Styling: Matplotlib offers extensive customization options to control every aspect of a plot's appearance, including colors, line styles, markers, labels, titles, axes, and annotations. Users can customize plots using Python scripting or interactively using Matplotlib's interactive interfaces like Matplotlib.pyplot and Object-Oriented API.

- Integration with Pandas: Matplotlib integrates seamlessly with Pandas, making it easy to visualize Pandas DataFrames and Series. Users can directly plot data from Pandas objects using Matplotlib's plotting functions, leveraging the flexibility and power of both libraries to create insightful visualizations from tabular data.

- Support for Multiple Output Formats: Matplotlib supports multiple output formats, allowing users to save plots in various file formats such as PNG, PDF, SVG, and more. Additionally, Matplotlib provides interactive backend options for embedding plots in graphical user interfaces (GUIs) or web applications, enabling users to create interactive visualizations for exploratory data analysis or presentations.

LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 Literature Study

1. APPLICATION OF SOFTWARE ENGINEERING IN STUDENT TIME MANAGEMENT USING PROTOTYPE MODEL

Exaudina Glory Sianturi , Sharon CedilaSuryadi, Pascal Wilman, Maria Susan Anggreainy, Syaeful Karim, Harvianto Computer Science Department School of Computer Science Bina Nusantara University Jakarta, Indonesia.

Time management is one of the most important things in humankinds' life, especially in student activity. In the new pandemic era, students have already changed and adjust their learning method to an online learning system which does not need them to have activities outside their house. This condition might make them have a lot of time but they lack the ability to manage their own study schedule. To handle this situation, we construct and develop a scheduler application. This application uses Prototyping Model as the development method as it is a user-oriented application. We are using a prototyping model to consider the upcoming details required in the development. It involves communication between user and developer continuously to achieve the goals and things needed by the user. The implementation of software engineering in student time management with a prototype model will upgrade students' skill to the next level of the digital era.

Summary:

This introduces a scheduler application, My Selvis, designed to aid students in managing their time effectively amidst the shift to online learning during the Covid-19 pandemic. Utilizing the Prototyping Model, the application aims to address procrastination and enhance time management skills through continuous user feedback. Key features include Today's, Schedule, and Collaboration features, with a focus on providing notifications to users for schedule management. Testing methods such as the Eight Golden Rules ensure usability and effectiveness. Future plans involve integration with Google Calendar for enhanced functionality. The research is supported by Bina Nusantara University's International Research Grant.

2. AUTOMATED TIME MANAGER: EFFECTIVENESS OF SELF-REGULATION ON TIME MANAGEMENT THROUGH A SMARTPHONE APPLICATION.

Bogoan Kim , Seok-Won Lee ,Hwajung Hong , and Kyungsik Han Department of Software and Computer Engineering, Ajou University, Republic of Korea Department of Communication, Seoul National University, Republic of Korea We investigated the effectiveness of a self-regulation strategy on time management leveraged by smartphone capabilities using a theoretical framework of self-regulation that consists of four elements: (a) goal setting, (b) task strategy utilization, (c) self-monitoring and reflection, and (d) self-efficacy and intrinsic motivation. We determined goals and strategies adopted during college life by surveying 295 college students and identified time management as a fundamental element for achieving such goals and strategies. To improve students' time management, we developed a smartphone application, Automated Time Manager (ATM), designed to provide users with visualizations of their physical activities and phone usage reports and also to acquire smartphone sensor and usage data. From a field study of 46 college students, we highlighted three primary user experiences – awareness of unawareness, preferred feedback, contextual but obvious use – and an overall positive time management outcome with ATM. We present an empirical study that transforms self-regulation, a well-known approach in social sciences, into computing, and discuss the salient design implications for supporting time management in a more effective manner with a smartphone application.

Summary:

Although students' excessive use of smartphones is indeed a problem in our society, we need to acknowledge that students' smartphone use is inevitable; therefore, restriction on its use would not be a good solution. Prior work has emphasized the key role that self-regulation of behaviour changes plays in achieving positive outcomes and experiences. Our research is motivated by the lack of studies on students' self-regulation of smartphone usages and is grounded by a theoretical framework to support such a psychological aspect. We identified time management as a common strategy for students to achieve their goals by a bottom-up approach, and developed a mobile application, ATM, to support self-control and self-reflection aspects of self-regulation. As a result of the user study, we identified three findings: awareness of unawareness, preferred feedback, and contextual but obvious use. Lastly, we presented future design

implications for supporting self-regulation of time management in a more effective fashion.

3. SCHEDULEME - SMART DIGITAL PERSONAL ASSISTANT FOR AUTOMATIC PRIORITY BASED TASK SCHEDULING AND TIME MANAGEMENT.

At present, it has become challenging for university students to manage their workload such assignments, projects etc. among their day-to-day tasks and personal chores. It has become hard to spend time efficiently on tasks that should be prioritized, and to decide what the best way to spend their remaining time is. Even though integration methods and multi-functional Time Management Tools (TMTs) such as Trello and Asana exist, finding, following, and implementing them is time consuming and monotonous. ScheduleME is a smart digital personal assistant, which will be in a form of a mobile app that collects and stores all the tasks the student must do, prioritize them according to their importance, schedule them intelligently across the student's remaining time considering his/her existing academic and personal timetables and daily routines. A user-friendly and comprehensible mobile app is designed where the right amount of information is presented to the user without important details that user could configure and override and not show too much information such that the user becomes overwhelmed. This overcomes the weakness found in many time-management and to do list apps. (e.g. - Trello, Microsoft Tasks, Todoist) where the user must enter all the details of the tasks manually and set the priority manually. The main emphasis of our suggested system is four primary components. They are Data engineering, Intelligent task breakdown and scheduling, Personalized task scheduling and User-cantered interaction design. Aside from that, this system employs a variety of technologies and algorithms to improve the research's accuracy and efficiency.

Summary:

University students face increasing time management challenges due to diverse tasks like online education, social networking, and fitness management. This research targets undergraduates at SLIIT, Sri Lanka, addressing the lack of tailored time management tools. The proposed system automates data scraping from Courseweb and allocates tasks based on remaining time, using constraint programming. Additionally, reinforcement learning suggests optimal task scheduling, while user-cantered design

ensures simplicity. Existing tools like time trackers and note takers have limitations in prioritization and expressiveness. The proposed system overcomes these shortcomings, providing a smart digital assistant for effective time management. Evaluation shows significant accuracy, filling a gap in existing tools and research, catering to the needs of reactive and unorganized students. Overall, ScheduleME enhances time management for university students, considering portability, accessibility, and security concerns.

2.2 EXISTING SYSTEM:

A simple to do list application where we specify the *task names* and the *time* to do them. And it notifies the user based on the time given by the user. Marking the status of the task whether done or not done. The existing system does not have the capability to specify and manage the priorities. It does not have flexibility to reschedule the tasks if the user skips the task on time. There is no mention of time bound and unbound tasks. Time bound tasks are the tasks which must be performed in the specified time interval. While the time bound tasks can be performed by the user in their free time without being confined to the time interval.

2.3 PROPOSED SYSTEM:

The main goal of our proposed system was to reduce the stress by doing the right things at right time. It has the capability to specify and manage the priorities implicitly without being explicitly specified. It has flexibility to reschedule the tasks if the user skips the task. On observing the user behaviour of skipping and doing tasks the application will recommend the tasks based on users' interest and priorities using reinforcement learning algorithms. A new feature of achieving goals on daily basis is added that is, suppose you add a goal like learning JavaScript course, no. of days to complete and hours you spend daily. The application will intelligently schedule so that you won't miss it.

2.4 FEASIBILITY STUDY:

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are.

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

2.4.1 ECONOMICALFEASIBILITY:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour while implementing an Actor-Critic Reinforcement Learning-based time management system entails initial investment costs, the potential benefits in terms of increased productivity, time savings, and improved performance make it economically feasible, especially in today's fast-paced digital landscape where efficient time management is paramount.

2.4.2 TECHNICALFEASIBILITY:

This study is carried out to check the economic impact that the system will have on the organization. The technical feasibility of an Actor-Critic Reinforcement Learning-based time management system hinges on the availability of suitable data, computational resources, algorithmic sophistication, integration capabilities, and adherence to privacy and security standards. With careful planning, implementation, and testing, such a system can offer a technically viable solution for addressing the challenges of time management in the digital era.

2.4.3 SOCIAL FEASIBILITY:

This study is carried out to check the economic impact that the system will have on the organization. The social feasibility of an Actor-Critic Reinforcement Learning-based time management system depends on its ability to address users' needs, enhance productivity and well-being, respect cultural and ethical considerations, promote inclusivity, and foster community engagement. With careful planning, stakeholder involvement, and sensitivity to societal dynamics, the proposed system can contribute positively to individuals' lives and broader societal goals.

ANALYSIS AND DESIGN

3.1 REQUIREMENTS

SOFTWARE REQUIREMENTS

Software Requirements is a field within software engineering that deals with establishing the needs of stakeholders that are to be solved by software. The IEEE Standard Glossary of Software Engineering Terminology defines a requirement as:

- A condition or capability needed by a user to solve a problem or achieve an objective.
- A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document.
- A documented representation of a condition or capability as in 1 or 2.

The activities related to working with software requirements can broadly be broken down into elicitation, analysis, specification, and management.

The software requirements are a description of features and functionalities of the target system. The requirements can be obvious or hidden, known or unknown, expected, or unexpected from the client's point of view. The process to gather the software requirements from clients, analyse and document them is known as software requirement analysis.

USER REQUIREMENTS

1. **Access to a compatible device**: The user will need to have a compatible Android device that meets the app's hardware and software requirements.
2. **Knowledge of seed identification**: The user should be able to correctly identify the type of seed they are testing to ensure that the app provides accurate analysis and information.
3. **Good lighting conditions**: The user will need to take clear pictures of the seeds with good lighting conditions to ensure that the app can analyse the images effectively.
4. **A steady hand**: The user should be able to hold the device steady while taking pictures of the seeds to avoid blurry images and inaccurate analysis
5. **Proper positioning of the seeds**: The user will need to ensure that the seeds are properly positioned in front of the camera lens to ensure accurate analysis.

3.1.1 FUNCTIONAL REQUIREMENTS

In Software engineering and systems engineering, a functional requirement defines a function of a system or its component. A function is described as a set of inputs, the behaviour, and outputs. Functional user requirements may be high-level statements of what the system should do but functional system requirements should also describe clearly the system services in detail. The requirements in requirement engineering help direct the development of the engineered product.

User Registration and Profile Management:

- Users should be able to register accounts and create profiles.
- Profiles should include information such as personal preferences, goals, and past performance.

Task Management:

- Users should be able to input tasks with details such as deadlines, priority levels, and estimated durations.
- The system should provide functionality for organizing, updating, and deleting tasks.

Reinforcement Learning Model Integration:

- Integration of an Actor-Critic reinforcement learning model into the time management system.
- The model should continuously learn from user behaviour and feedback to adapt its recommendations.

Personalized Time Management Strategies:

- The system should generate personalized time management strategies based on individual behaviours and preferences.
- Strategies should prioritize tasks, allocate time slots, and adjust schedules dynamically.

Dynamic Learning and Adaptation:

- Continuous learning and adaptation of the reinforcement learning model based on user interactions.
- The system should update recommendations in real-time as user behaviour changes.

User Feedback Mechanism:

- Users should be able to provide feedback on suggested time management strategies.
- The system should use feedback to improve future recommendations and adaptability.

Visualization and Reporting:

- Visual representations of task schedules, priorities, and time allocations.
- Reporting functionality to track productivity metrics, progress towards goals, and overall effectiveness.

Accessibility and Usability:

- Intuitive user interface design for easy navigation and interaction.
- Accessibility features to accommodate users with different needs and preferences.

Privacy and Security:

- Implementation of measures to ensure the privacy and security of user data.
- Compliance with relevant data protection regulations and best practices.

Integration with External Tools:

- Integration with calendar applications, to-do list apps, or project management software for seamless task synchronization.
- APIs or plugging to connect with third-party services commonly used for time management.

3.1.2 NON-FUNCTIONAL REQUIREMENTS

Performance:

- The system should respond to user interactions quickly, with minimal latency.
- It should be able to handle a large number of concurrent users without significant degradation in performance.

Scalability:

- The system should be designed to scale horizontally to accommodate increasing user loads.
- It should be able to add additional resources seamlessly to handle increased demand.

Reliability:

- The system should have high availability, with minimal downtime for maintenance or unexpected issues.
- It should be resilient to failures, with built-in redundancy and failover mechanisms.

Accuracy:

- The recommendations generated by the reinforcement learning model should be accurate and relevant to user needs.
- The system should minimize errors in task scheduling and time allocation.

Security:

- Data transmission and storage should be encrypted to protect user privacy.
- Access controls should be implemented to ensure that only authorized users can view or modify sensitive information.
- Regular security audits and updates should be conducted to identify and address potential vulnerabilities.

Usability:

- The user interface should be intuitive and easy to navigate, with clear instructions and feedback.

- Accessibility features such as screen readers and keyboard navigation should be provided to accommodate users with disabilities.

Compatibility:

- The system should be compatible with a wide range of devices and web browsers to ensure accessibility for all users.
- It should also be compatible with different operating systems and versions.

Maintainability:

- The system should be well-documented to facilitate maintenance and future development.
- Code should be modular and organized, making it easy to update or extend functionality.

Regulatory Compliance:

- The system should comply with relevant data protection regulations, such as GDPR or CCPA.
- It should also adhere to industry standards and best practices for time management software.

Performance Metrics:

- Metrics should be defined and tracked to measure the system's performance, including response time, uptime, and user satisfaction.
- Regular monitoring and analysis should be performed to identify areas for improvement.

3.2 SYSTEM SPECIFICATIONS

Hardware Requirements:

Server:

- Processor: Multi-core processor with sufficient processing power to handle concurrent user requests.
- Memory: At least 8GB of RAM for optimal performance.
- Storage: SSD storage for fast read/write operations.

Client Devices:

Desktops, Laptops, tablets, and smartphones with modern web browsers.

Software Requirements:

- Operating System: Linux-based server operating system (e.g., Ubuntu Server) for hosting the application.
- Web Server: Apache or Nginx for serving web pages and handling HTTP requests.
- Database Management System: PostgreSQL or MySQL for storing user data and task information.
- Programming Languages: Python for backend development, HTML, CSS, and JavaScript for frontend development.

Frameworks and Libraries:

- Backend: Flask or Django for web application development.
- Frontend: React.js or Vue.js for building dynamic user interfaces.
- Reinforcement Learning: TensorFlow or PyTorch for implementing the Actor-Critic model.
- Version Control: Git for managing codebase changes and collaboration.
- Containerization: Docker for packaging the application and its dependencies into containers.
- Orchestration: Kubernetes for container orchestration and management in production environments.

Network Requirements:

- Internet connectivity for users to access the application.
- Secure HTTPS protocol for data encryption and secure communication between clients and the server.
- Load balancing and CDN (Content Delivery Network) for distributing traffic and optimizing performance.

Security Requirements:

- Secure Socket Layer (SSL) certificate for encrypting data transmitted over the network.
- Authentication and Authorization: Implement user authentication mechanisms (e.g., username/password, OAuth) to control access to the system.
- Data Encryption: Encrypt sensitive user data stored in the database.
- Regular security updates and patches to address vulnerabilities and mitigate security risks.
- Backup and Recovery: Automated backup mechanisms to regularly backup user data and system configurations.
- Disaster recovery plan in place to restore the system in case of data loss or system failure.

Scalability and Performance:

- Horizontal Scaling: Design the system to scale horizontally by adding more servers to handle increasing user loads.
- Performance Monitoring: Implement monitoring tools to track system performance metrics (e.g., CPU usage, memory usage, response time) and identify performance bottlenecks.

Load testing to assess the system's performance under different traffic conditions and optimize resource allocation.

These system specifications outline the hardware, software, network, security, and scalability requirements necessary to deploy and operate the time management system effectively. Actual specifications may vary based on specific deployment environments and performance requirements.

3.4 UML DIAGRAMS

Unified Modeling Language (UML) is a general-purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite like blueprints used in other fields of engineering. UML is linked with object-oriented design and analysis. UML makes the use of elements and

forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

1. Structural Diagrams– Capture static aspects or structure of a system. Structural Diagrams include Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.

2. Behavior Diagram– Capture dynamic aspects or behavior of the system. Behavior diagrams include Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.

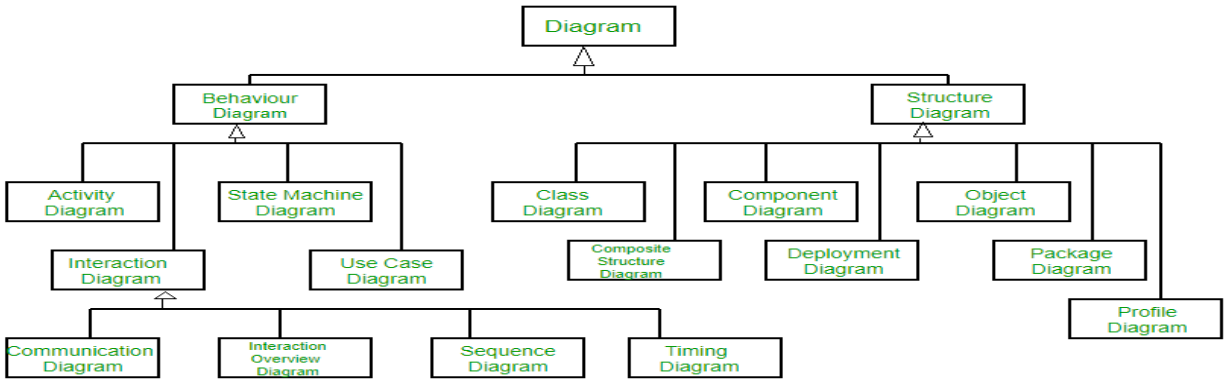


Figure 3.2:UML Diagrams Classification

3.4.1 Use Case Diagram:

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and tells how the user handles a system.

For this system, as shown in the below diagram there will be ten use cases and 2 actors. This can depict various use cases such as setting goals, creating tasks, tracking progress, receiving recommendations, and providing feedback. It shows the actors and their interactions with the system, highlighting the specific actions and outcomes associated with each use case.

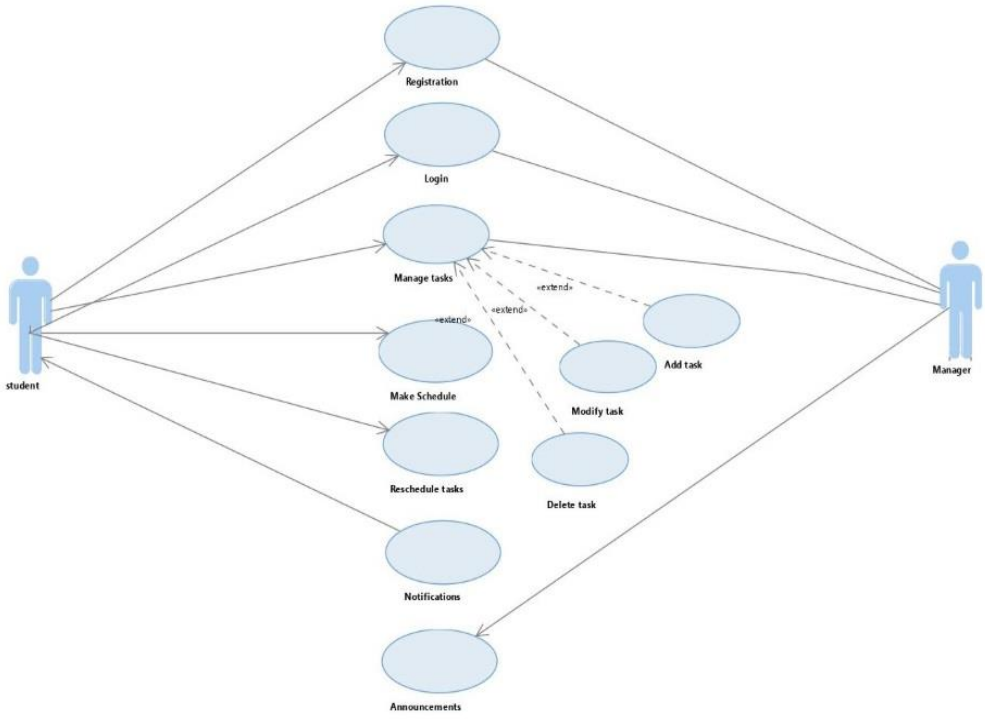


Figure 3.3: Use Case Diagram

3.4.2 Class Diagram:

The class diagram depicts a static view of an application. It represents the types of objects residing in the system and the relationships between them. A class consists of its objects, and it may inherit from other classes. A class diagram is used to visualize, describe, document various aspects of the system, and construct executable software code. In the class diagram, we represented three classes with several operations or functions.

This class helps organize tasks, deadlines, priorities, and methods for adding tasks and setting deadlines. By utilizing a class, you can structure and manage your time-related information effectively. This allows for seamless integration with reinforcement learning algorithms, which can optimize your time management decisions based on the agent's interactions with the environment.

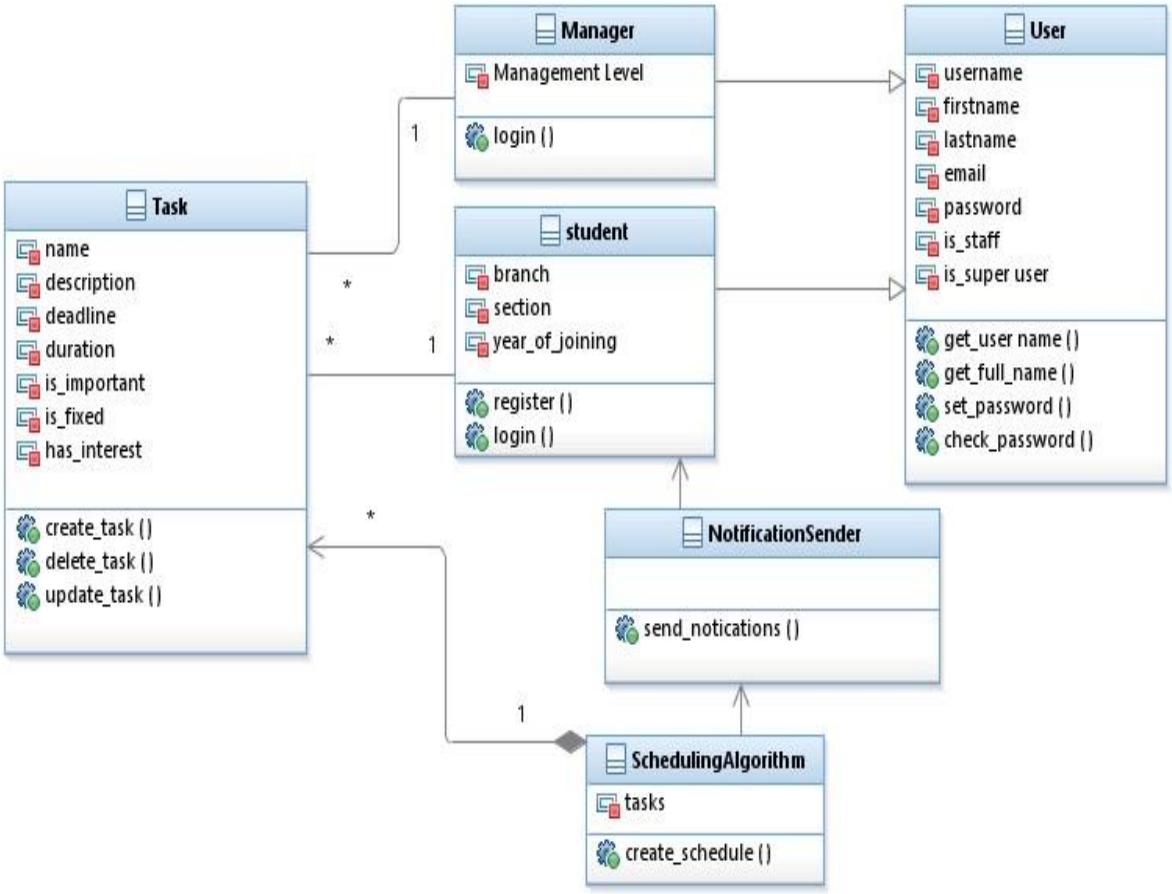


Figure 3.4: Class Diagram

3.4.3 Sequence Diagram:

3.4.3.1 Sequence Diagram for registration:

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as b ranching. In the sequence diagram, we took two lifelines user and admin.

The diagram captures the steps involved in the registration process, starting with the user accessing the registration page. The user then provides the required information, such as their name, email address, and password. The system processes this information, validating and storing it securely. The sequence diagram can also include steps like checking for duplicate accounts, sending a verification email, and providing confirmation to the user once the registration is successful.

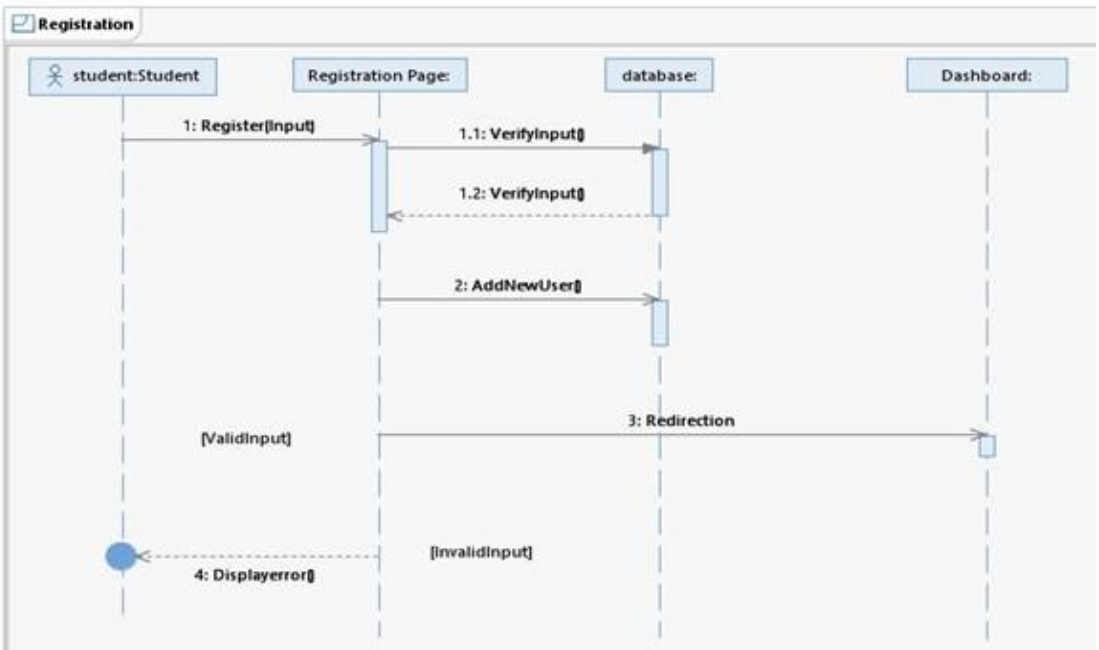


Figure 3.5: Sequence Diagram for registration page

3.4.3.2 Sequence Diagram for login page:

The role of the diagram is to illustrate the sequence of interactions between the user and the login system. It shows the steps involved in the login process, such as entering credentials, verifying them, and granting access. The sequence diagram helps visualize the flow of information and actions between the user and the login system, aiding in understanding and optimizing the login functionality.

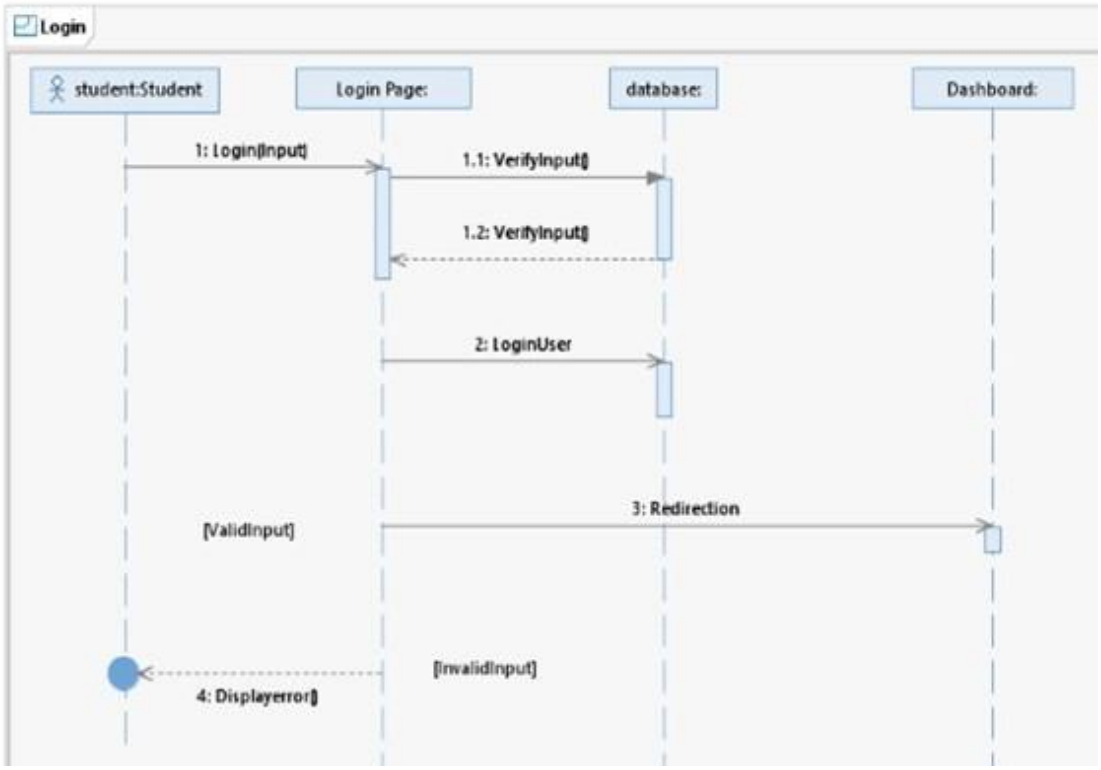


Figure 3.6: Sequence Diagram for login page

3.4.3.3 Sequence Diagram for adding tasks:

The sequence diagram represents the adding a task in a time management system, the diagram illustrates the step-by-step process involved. It begins with the user initiating the task addition process through the user interface. The user then provides the necessary task details, such as the task name, due date, and priority level. The system processes this information, ensuring its validity and accuracy. Once validated, the system updates the task list by adding the new task. Finally, the system provides confirmation or feedback to the user, indicating the successful addition of the task. By visualizing this sequence of interactions, the diagram helps in understanding and optimizing the task addition process for efficient time management in the digital era.

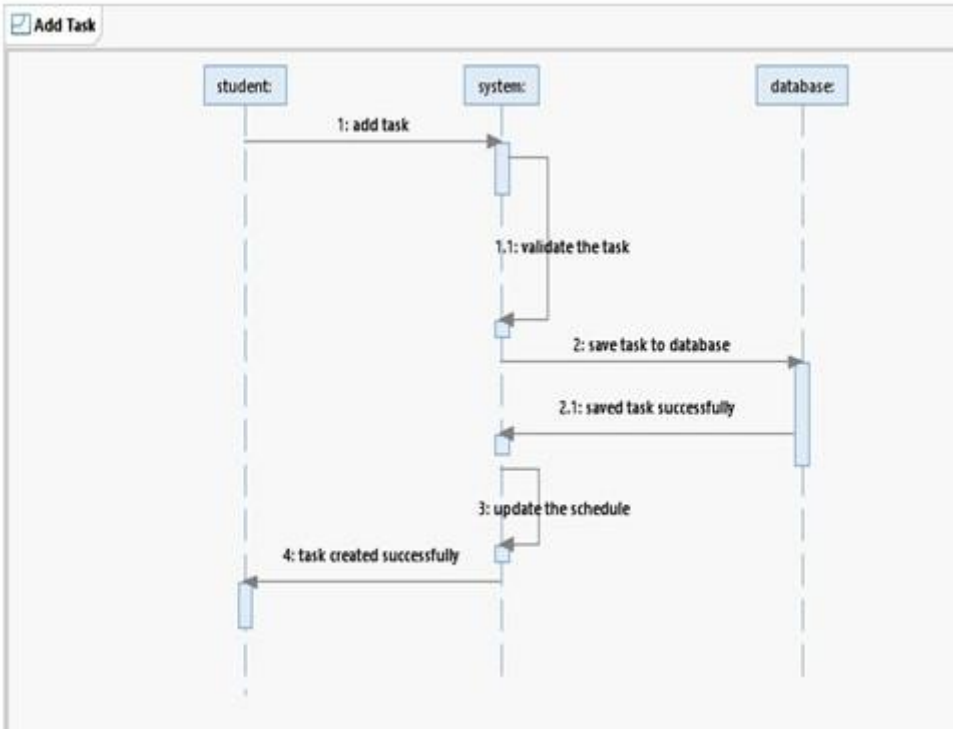


Figure 3.7: Sequence Diagram for adding tasks

3.4.3.4 Sequence Diagram for manage tasks:

The sequence diagram for managing tasks is the interactions between the user and the system when performing various tasks management actions. This can include actions such as viewing tasks, editing task details, marking tasks as completed, deleting tasks, and any other relevant task management actions. The sequence diagram helps visualize the flow of information and actions, aiding in understanding and optimizing the task management process for efficient time management in the digital era.

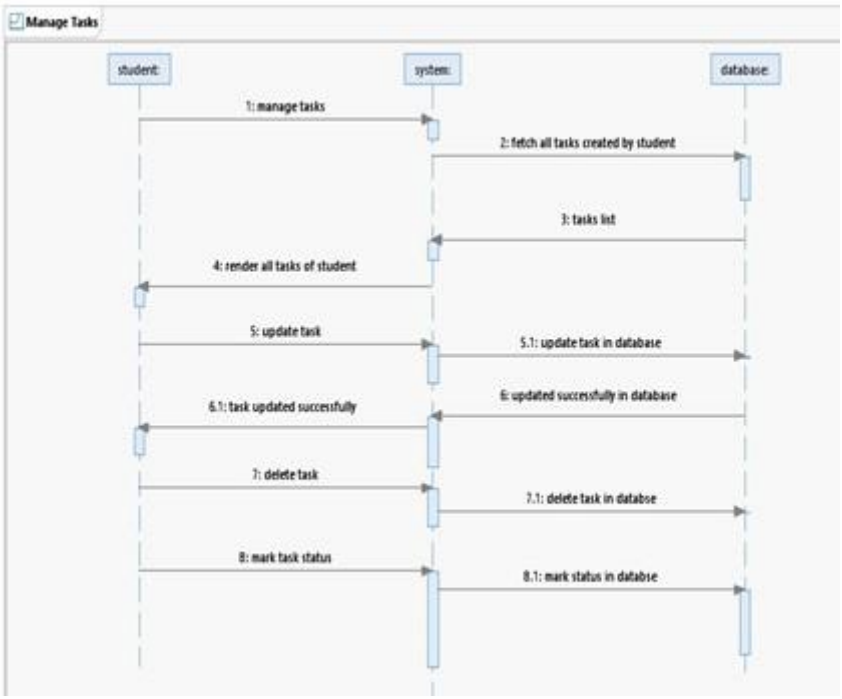


Figure 3.8: Sequence Diagram for managing the tasks

3.4.3.4 Sequence Diagram for reschedule the tasks:

The sequence diagram rescheduling tasks is to illustrate the sequence of interactions between the user and the system when modifying the schedule or due dates of tasks in a time management system. The diagram captures the steps involved in rescheduling tasks, starting with the user selecting a task to be rescheduled. The user then provides the updated due date or new schedule for the task. The system processes this information, updating the task's schedule accordingly. The sequence diagram can also include steps like checking for conflicts with other tasks, notifying the user of the successful rescheduling, and updating the task list or calendar view.

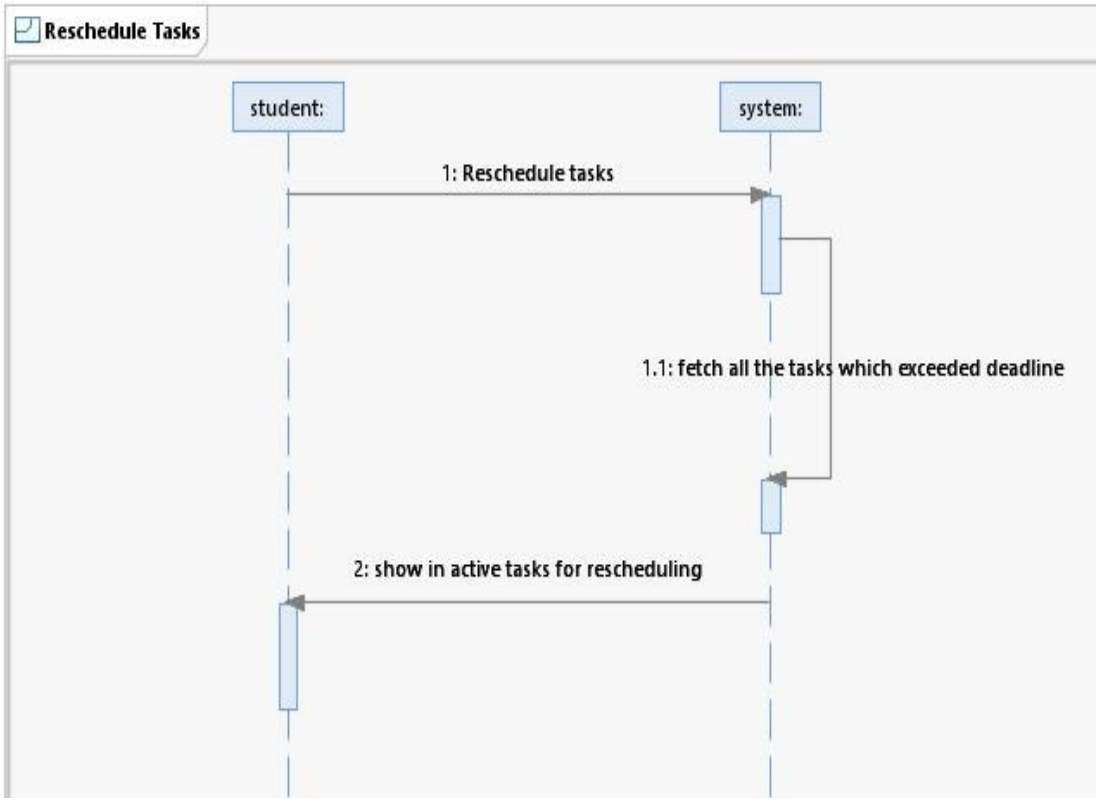


Figure 3.9: Sequence Diagram for rescheduling the tasks

3.4.4 Collaboration Diagram:

Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming. An object consists of Several features. Multiple objects present in the system are connected to each other. The Collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

Notation

1. Objects
2. Actors
3. Links
4. Messages

As we know, sequence diagram is like collaboration diagram, but lifelines will not be considered. The lifelines in sequence diagram will be taken as objects in collaboration diagram. The same timeline of messages and interaction is followed in collaboration diagram as in sequence diagram. In this, a collaboration diagram can depict how different components, such as the user interface, task management module, notification system, and database, work together to achieve specific functionalities. It shows the relationships, dependencies, and communication channels between these components, helping to visualize the overall system architecture and how different parts of the system collaborate to perform tasks and deliver features.

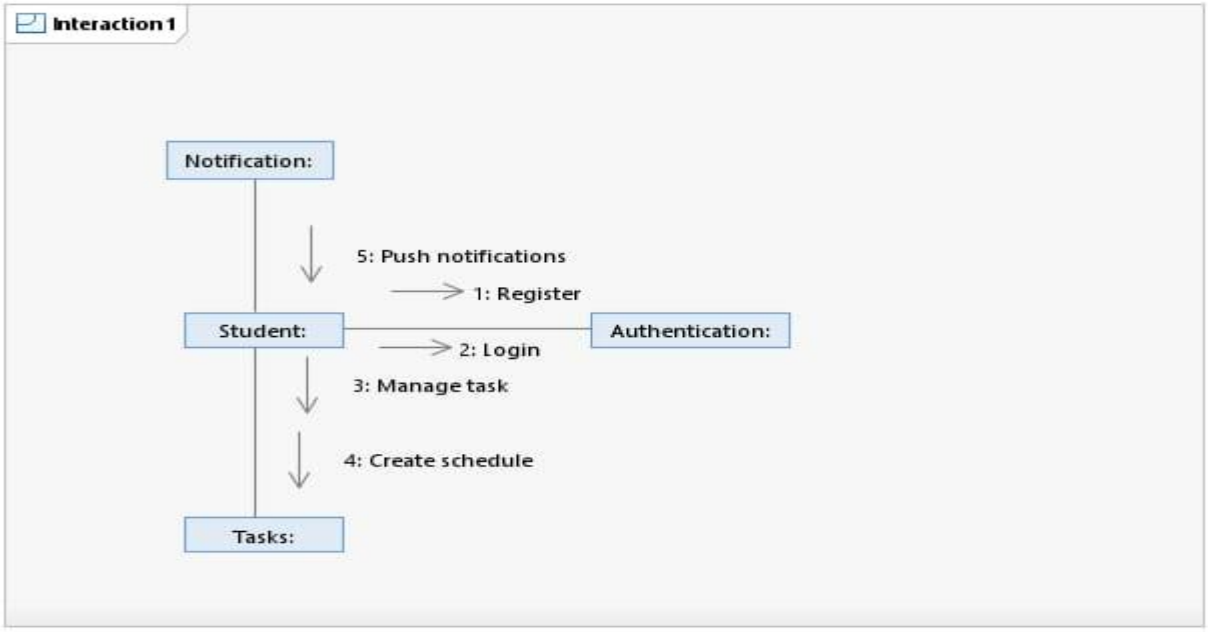


Figure 3.10: Collaboration Diagram

3.4.6 Component Diagram:

A component diagram is used to break down a large object-oriented system into the smaller components, to make them more manageable. It models the physical view of a system such as executables, files, libraries, etc. that resides within the node. It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system.

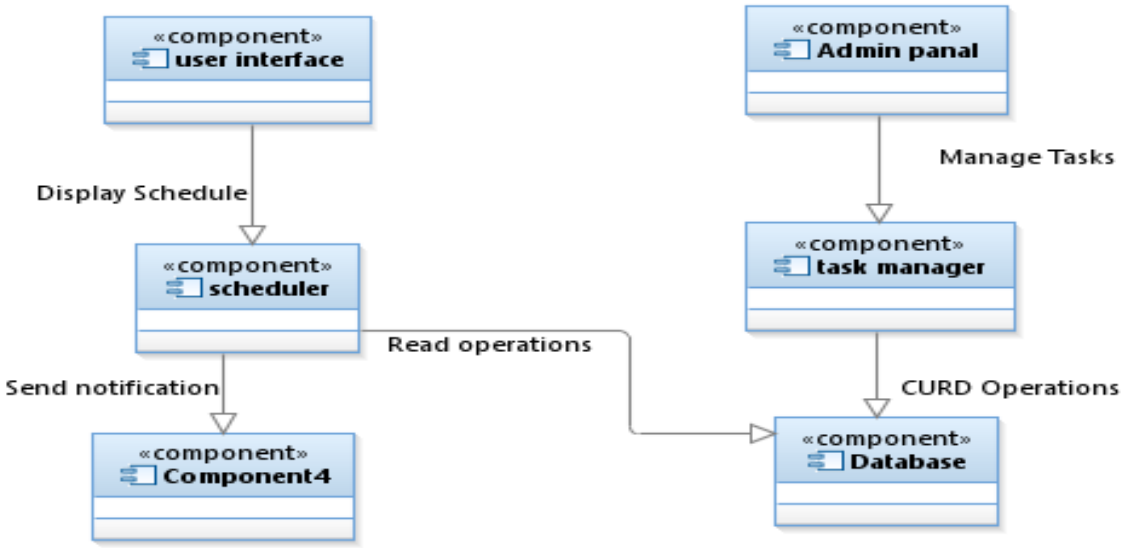


Figure 3.12: Component Diagram

IMPLEMENTAION

4.1 Task object details

This section provides an in-depth exploration of the task objects utilized in scheduling system. These objects include crucial data about tasks and their interaction with the reinforcement learning system to optimize scheduling.

Essential Attributes for Scheduling

- schedule_after (datetime): This attribute represents the earliest permissible time to start working on a task. It corresponds to the release time (t_release) in scheduling terminology. This ensures the task isn't scheduled before a specific point in time.
- duration (timedelta): This attribute defines the amount of time required to complete the task. It's crucial for scheduling as it determines the length of the time slot needed to allocate for the task. The scheduler considers this duration when fitting tasks into the schedule.
- deadline (datetime): This attribute specifies the target completion time for the task (t_drop). It helps prioritize tasks and schedule them considering their urgency. Tasks approaching their deadlines might be prioritized over tasks with more lenient deadlines.

Additional User-oriented Attributes

- name (string): This attribute provides a human-readable name for the task. This makes it easier to identify and interact with tasks within the system.
- description (string): A description of the task can be provided in this attribute. This additional context helps the user understand the nature of the task.
- is_important (bool): This Boolean flag indicates the user's perceived importance of the task. The scheduler can leverage this information to prioritize tasks the user considers critical.
- has_interest (bool): This Boolean flag indicates the user's level of interest in performing the task. While not directly related to scheduling, this information could be used by the system to suggest tasks the user might find more engaging, potentially improving adherence to the generated schedule.

Reinforcement Learning Integration

- call method: This method implements a loss function that quantifies the penalty associated with delayed task execution. This function plays a vital role in reinforcement learning. It provides feedback to the learning algorithm by indicating how well the scheduling decisions are performing. The loss function being monotonic non-decreasing ensures that the penalty increases as the task deviates further from its ideal deadline. This guides the learning algorithm to favour schedules that minimize these penalties, leading to improved task completion within deadlines.

Quantifying the Loss for Delayed Execution

In our time management system with reinforcement learning, effectively quantifying the loss for delayed tasks is crucial. This loss function guides the learning algorithm towards scheduling decisions that minimize delays and ensure timely task completion. Here's a breakdown of how different components contribute to this quantification.

Task Object Attributes

Deadline (t_drop): This remains the core element, representing the target completion time for the task. It serves as the benchmark against which delays are measured.

Additional Attributes for Loss Function:

- Slope: This attribute defines the rate at which the loss increases for delays before the deadline (t_drop). A higher slope indicates a steeper penalty as the task gets closer to its deadline. This allows the system to prioritize tasks with stricter deadlines and potentially nudge the user to complete them sooner.
- l_drop (Drop Loss): This attribute represents the additional loss incurred after the deadline (t_drop) has passed. It essentially defines the severity of the penalty for missing the deadline. A higher l_drop signifies a more significant consequence for delays beyond the deadline.

The task object can implement a __call__ method that takes the actual completion time as input and returns the calculated loss based on the attributes mentioned above.

This method can calculate the loss based on the following logic:

- If the task is completed before the deadline, there's no loss.
- If the task is completed before a specific time point within the deadline window (defined by t_drop), the loss increases linearly with the delay, determined by the slope.
- If the task is completed after the t_drop time, a sharp increase in loss occurs represented by l_drop, followed by a continued linear increase based on the slope.

Benefits of this approach:

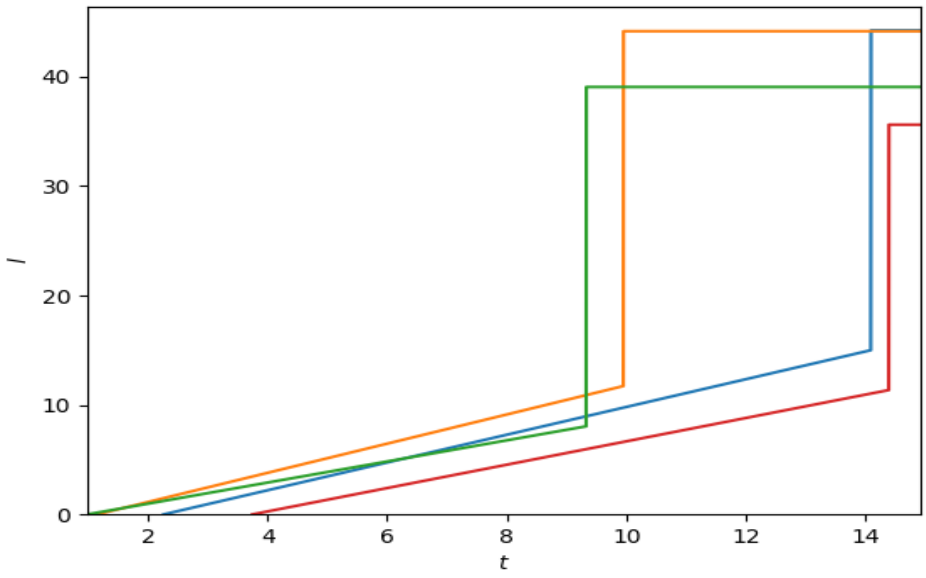
- Prioritization: Tasks with stricter deadlines and higher slopes experience a steeper rise in loss as they approach their deadlines, prompting the scheduler to prioritize them.
- Flexibility: Different slopes and l_drop values can be assigned to tasks based on their importance and the consequences of missing deadlines.
- Learning Algorithm Guidance: The loss function provides clear feedback to the reinforcement learning algorithm, guiding it towards scheduling decisions that minimize overall loss and ensure timely task completions.

Additional Considerations:

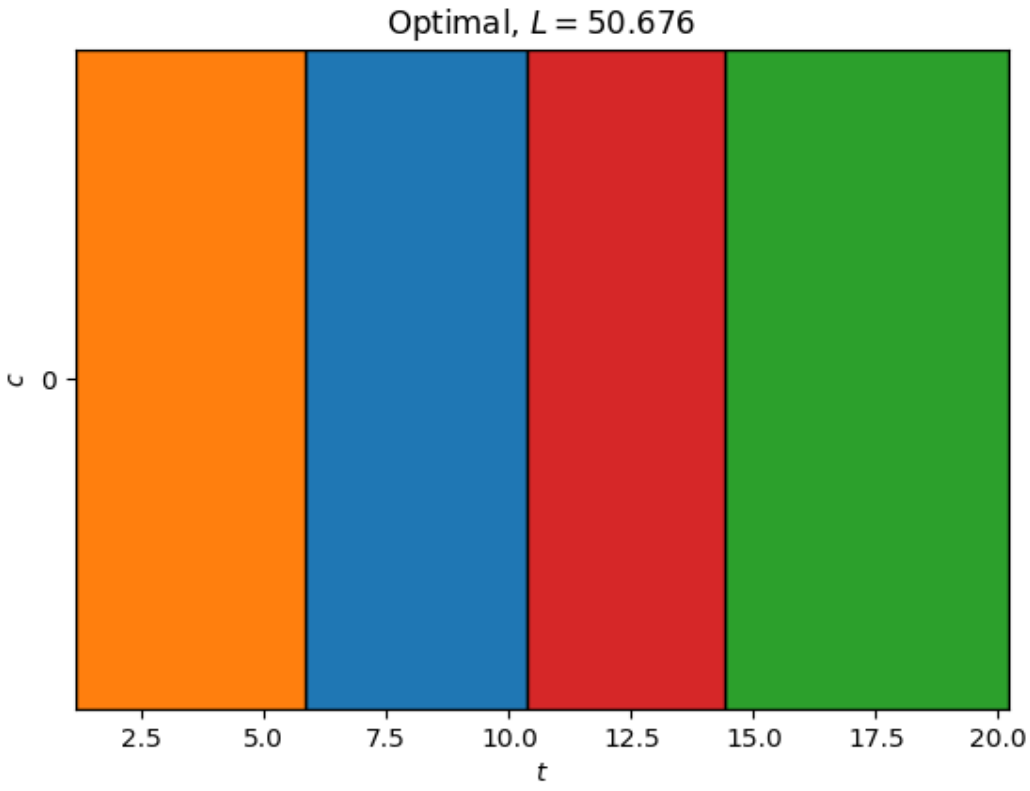
- While this approach focuses on deadlines, incorporating additional factors like task importance or user interest could further refine the loss function.
- Experimenting with different weighting schemes for slope, l_drop, and other potential factors can help optimize the system's performance for specific use cases.

By effectively quantifying the loss for delayed execution, your time management system can learn to create personalized schedules that consider both task urgency and the consequences of missing deadlines, ultimately leading to improved user productivity and goal achievement.

Tasks – time vs. loss for delayed execution



Optimal Schedule



4.2 ACTOR CRITIC ALGORITHM

The actor-critic algorithm is a type of reinforcement learning algorithm that combines aspects of both policy-based methods (Actor) and value-based methods (Critic). This hybrid approach is designed to address the limitations of each method when used individually.

In the actor-critic framework, an agent (the “actor”) learns a policy to make decisions, and a value function (the “Critic”) evaluates the actions taken by the Actor.

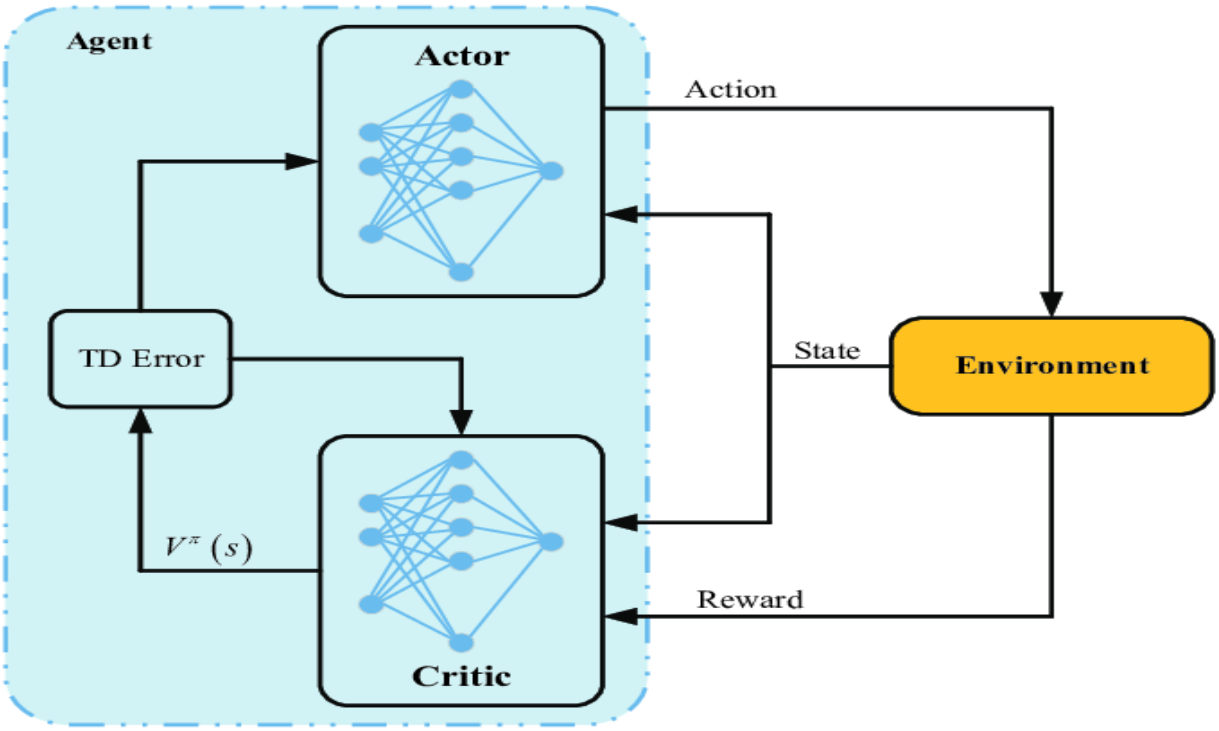
Simultaneously, the critic evaluates these actions by estimating their value or quality. This dual role allows the method to strike a balance between exploration and exploitation, leveraging the strengths of both policy and value functions.

Actor Critic is a reinforcement learning algorithm that combines the advantages of policy gradient and value-based methods.

Actor Critic uses two neural networks:

- The actor network predicts the probability of taking an action in a given state.
- The critic network estimates the value of a state-action pair.

Actor Critic works by iteratively updating the actor and critic networks. The actor network is updated using policy gradient, which means that it is updated in the direction that increases the expected reward. The critic network is updated using temporal difference learning, which means that it is updated to estimate the value of the state-action pairs that the actor network has taken.



Advantage function for actor-critic:

$$A_{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t)$$

Alternatively, advantage function is called as TD error as shown in the Actor-Critic framework. As mentioned above, the learning of the actor is based on policy-gradient. The policy gradient expression of the actor as shown below:

$$\nabla J(\theta) \approx \sum_{t=0}^{T-1} \nabla_\theta log\pi_\theta(a_t, s_t) A_{\pi_\theta}(s_t, a_t)$$

Advantages of Actor Critic Algorithm

- ✓ **Improved Sample Efficiency**: The hybrid nature of Actor-Critic algorithms often leads to improved sample efficiency, requiring fewer interactions with the environment to achieve optimal performance.

- ✓ **Faster Convergence**: The method's ability to update both the policy and value function concurrently contributes to faster convergence during training, enabling quicker adaptation to the learning task.
- ✓ **Versatility Across Action Spaces**: Actor-Critic architectures can seamlessly handle both discrete and continuous action spaces, offering flexibility in addressing a wide range of RL problems.
- ✓ **Off-Policy Learning**: Learns from past experiences, even when not directly following the current policy.

4.3 Pseudocode of Actor-Critic algorithm

1. Sample {s, t, a_t}using the policy π(θ) from the actor-network.
2. Evaluate the advantage function A_t. It can be called as TD error δt. In Actor-critic algorithm, advantage function is produced by the critic-network.

$$A_{\pi_\theta}(s_t, a_t) = r(s_t, a_t) + V_{\pi_\theta}(s_{t+1}) - V_{\pi_\theta}(s_t)$$

3. Evaluate the gradient using the below expression:

$$\nabla J(\theta) \approx \sum_{t=0}^{T-1} \nabla_\theta log \pi_\theta(a_t, s_t) A_{\pi_\theta}(s_t, a_t)$$

4. Update the policy parameters, θ

$$\theta = \theta + \alpha \nabla J(\theta)$$

5. Update the weights of the critic-based value-based RL(Q-learning). δt is equivalent to advantage function.

$$w = w + \alpha \delta_t$$

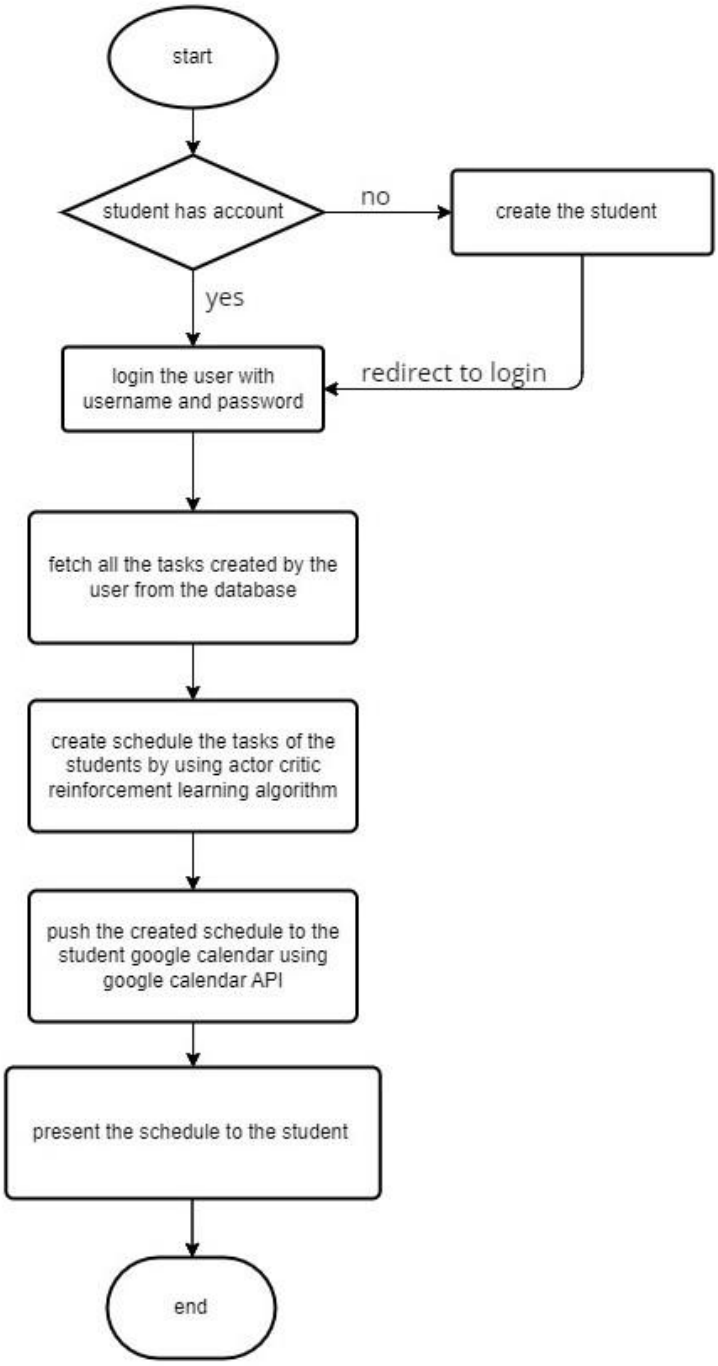
6. Repeat 1 to 5 until we find the optimal policy πθ.

4.4 Steps involved in the Actor-Critic algorithm

1. **Initialize Parameters**:
   - Initialize the parameters for both the actor (policy) and critic (value function).
2. **Environment Interaction**:
   - Start interacting with the environment.
   - Observe the current state s.
   - Perform an action a according to the actor's policy $\pi(a|s)$.
   - Receive the reward r and the next state s'.
3. **Critic Update**:
   - Compute the TD (Temporal Difference) error: $\delta = r + \gamma V(s') - V(s)$, where $V(s)$ is the value estimate of the current state.
   - Update the critic's parameters using the TD error: $V(s) \leftarrow V(s) + \alpha_v \delta$, where α_v is the critic's learning rate.

4. **Actor Update**:
   - Compute the advantage function: $A(s, a) = Q(s, a) - V(s)$, where $Q(s, a)$ is the estimated Q-value of taking action a in state s.
   - Update the actor's parameters using the advantage function gradient:
     $\nabla_{\theta_\pi} J(\theta_\pi) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_{\theta_\pi} \log \pi(a_i|s_i) A(s_i, a_i)$
     where θ_π are the parameters of the actor's policy, $\pi(a|s)$, and (s_i, a_i) are the states and actions sampled during the trajectory.
   - Update the actor's parameters using the gradient ascent rule: $\theta_\pi \leftarrow \theta_\pi + \alpha_\pi \nabla_{\theta_\pi} J(\theta_\pi)$, where α_π is the actor's learning rate.
5. **Repeat**:
   - Repeat steps 2-4 until convergence or a predefined stopping criterion is met.
6. **Termination**:
   - End the training process.

4.5 FLOW CHART



4.6 SOFTWARE INSTALLATION

Step 1: Install "Python"

1. Go to the official Python download page for Windows.
2. Find a stable Python 3 release version 3.9.
3. Click the appropriate link for your system to download the executable file: Windows installer (64-bit) or Windows installer (32-bit).
4. After the installer is downloaded, double-click the .exe file, for example python-3.9.18-amd64.exe, to run the Python installer.
5. Select the Install launcher for all user's checkbox, which enables all users of the computer to access the Python launcher application.
6. Select the Add python.exe to PATH checkbox, which enables users to launch Python from the command line.
7. Click Next.
8. Click Install to start the installation.
9. After the installation is complete, a Setup was successful message displays.
10. Enter the python --version command in the command prompt to verify the installation.

Step 2: Install "PostgreSQL"

1. Download Postgres Installer. Postgres Installer is available for PostgreSQL for various versions download version 16.
2. Click on the executable file to run the installer.
3. Select your preferred language.
4. Specify directory where you want to install PostgreSQL.
5. Specify PostgreSQL server port. You can leave this as default if you're unsure what to enter.
6. Specify data directory to initialize PostgreSQL database.
7. Create a PostgreSQL user password.
8. Create password for database Superuser.
9. Click next to begin PostgreSQL installation.
10. Close the installation window after completion.

Step 3: Create Virtual Environment

Go the project directory create the virtual environment using by running “python -m venv env_name”

Step 4: Install “Python Libraries using pip specified in requirements.txt”

- Go to the project directory and activate the environment by running source env_name/Scripts/activate for windows and source env_name/bin/activate for Linux.
- And run the command pip install -r requirements.txt to install all the dependencies for the project.

4.7 STEPS FOR EXECUTING THE PROJECT

Step 1: Switch to the project folder or directory in the visual code.

Step 2: Open the terminal or command prompt in the vs code.

Step 3: Create Django migrations using “python manage.py makemigrations”.

Step 4: Push the Django migrations to the PostgreSQL database with “python manage.py migrate” command.

Step 5: Create superuser for Django Administration by “python manage.py createsuperuser”, then fill username and password for the administrator account.

Step 6: Collect static files of Django Application with “python manage.py collectstatic”.

Step 7: Run the Django development server using “python manage.py runserver” command.

Step 8: Open the new terminal or command prompt in the visual studio code.

Step 9: Run celery my running “celery -A ProjectTime.celery worker -l info” in newly opened terminal.

Step 10: Switch back to the first terminal and copy the URL of the website and open it in the browser.

TESTING

5.1 TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are several types of tests. Each test type addresses a specific testing requirement.

5.1.1 TYPES OF TESTS

UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is cantered on the following items:

- Valid Input: identified classes of valid input must be accepted.
- Invalid Input: identified classes of invalid input must be rejected.
- Functions: identified functions must be exercised.
- Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

SYSTEM TESTING

System testing ensures that the entire integrated software system meets requirements. It evaluates a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

5.1.2 WHITE BOX TESTING

White Box Testing is a test in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to evaluate areas that cannot be reached from a black box level.

5.1.3 BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as

specification or requirements document, such as specification or requirements document. It is a test in which the software under the test is treated as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

5.1.4 LEVELS OF TESTING

5.1.4.1 UNIT TESTING:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is common for coding and unit testing to be conducted as two distinct phases. Test strategy and approach Field testing will be performed manually, and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link
- The entry screen, messages and responses must not be delayed

Features to be evaluated

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page

5.1.4.2 INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g., components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

5.1.4.3 ACCEPTANCE TESTING

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test ID	1
Name	Test case for registering to the application
Description	A user entire details were given in the registration page
Sample Input	Fill the details in the registration form
Expected Output	The information is stored and shown in the dashboard
Actual Input	The information is stored and shown in dashboard.

Test ID	2
Name	Test case for Login to the application
Description	A user enters details which were given in the registration page to login their dashboard
Sample Input	Enter the login details
Expected Output	The information which is stored will get login to the main dashboard
Actual Input	The information which is stored will get login to the main dashboard.

Test ID	3
Name	Test case for main dashboard to the application
Description	A user could find all the application domains on the dashboard to schedule the tasks
Sample Input	Enter the task details in the given dashboard
Expected Output	The scheduling task can enter the task according to the user credentials.
Actual Input	The scheduling task can enter the task according to the user credentials.

Test ID	4
Name	Test case for adding the task
Description	A user can give the task which he have to do by scheduling
Sample Input	Enter the task details in the given dashboard
Expected Output	Adding task can enter the task according to the user credentials which he must do by systematically.
Actual Input	Adding task can enter the task according to the user credentials which he must do by systematically.

Test ID	5
Name	Test case for task submission
Description	Submitting the task
Sample Input	Checking whether the task has been successfully added
Expected Output	Giving an input of task added successfully
Actual Input	Giving an input of task added successfully

Test ID	6
Name	Test case for managing the tasks.
Description	Managing the task or giving an info about whether the task had been completed or not.
Sample Input	Checking whether the task has been successfully completed
Expected Output	Giving an input of task has completed or to edit.
Actual Input	Giving an input of task has completed or to edit

Test ID	7
Name	Test case for rescheduling the task
Description	Incomplete task can be resheduled
Sample Input	The time ended tasks which are not attended can be rescheduled
Expected Output	The incomplete tasks can be resheduled
Actual Input	The incomplete tasks can be resheduled

Test ID	8
Name	Test case for connecting with Google calendar
Description	Connecting with Google calendar is to make sure to perform the tasks scheduled in the calendar application.
Sample Input	The scheduled tasks will be shown in the main application or in the format of message.
Expected Output	The scheduled tasks in the Google calendar will be placed in the main application.
Actual Input	The incomplete tasks in the Google calendar can be scheduled.

Test results: All the test cases mentioned above passed successfully. No defects encountered.

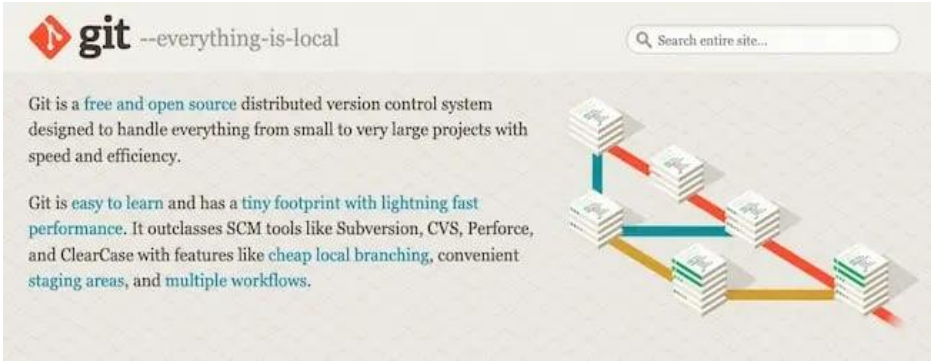
DEPLOYMENT

DEPLOYEMENT

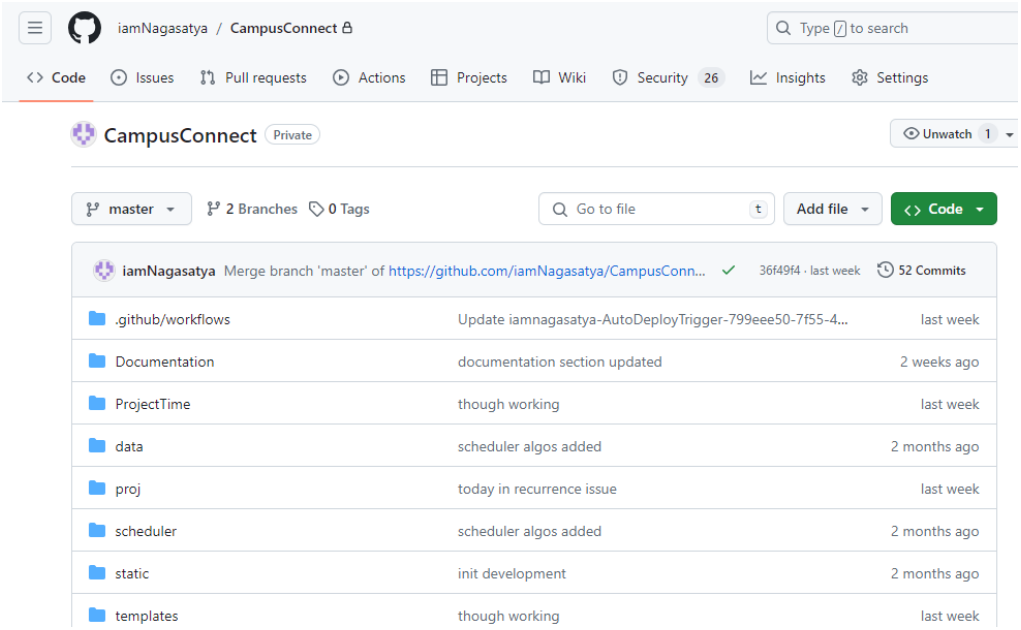
1. Git & GitHub

Git is open-source version control software, used for managing and tracking file revisions. You can use Git with any file type, but it's most often used for tracking code files. Git is open-source version control software, used for managing and tracking file revisions. You can use Git with any file type, but it's most often used for tracking code files

.



GitHub is a for-profit company that offers a cloud-based Git repository hosting service. Essentially, it makes it a lot easier for individuals and teams to use Git for version control and collaboration. GitHub's interface is user-friendly enough so even novice coders can take advantage of Git. Without GitHub, using Git generally requires a bit more technical savvy and use of the command line. Additionally, anyone can sign up and host a public code repository for free, which makes GitHub especially popular with open-source projects.



a. GitHub Workflows

GitHub workflows are automated processes that you can define within your GitHub repository. They are configured using YAML files (.yml) stored in the .github/workflows directory of your repository. These workflows can be triggered by various events, such as:

1. Code pushes (push)
2. Pull request creations or updates (pull_request)
3. Issue creations or updates (issues)
4. Schedule (e.g., run a workflow daily at a specific time)
5. Manually

Benefits of using GitHub workflows:

- Automation: Automates repetitive tasks, saving time and reducing manual errors.
- Continuous Integration (CI): Integrates code changes frequently, facilitating early detection of issues.
- Continuous Delivery/Deployment (CD): Automates building, testing, and deploying code, streamlining release processes.

- Improved Code Quality: Encourages code quality through automated testing and linting.
- Collaboration: Facilitates collaboration by ensuring everyone's code adheres to standards.
- Flexibility: Can be tailored to your specific project needs and preferences using YAML.

b. GitHub Actions

GitHub Actions is a built-in CI/CD (continuous integration and continuous delivery) platform that lets you automate your software development workflows directly within GitHub. It streamlines the process from code commits to production deployment, saving you time and effort.

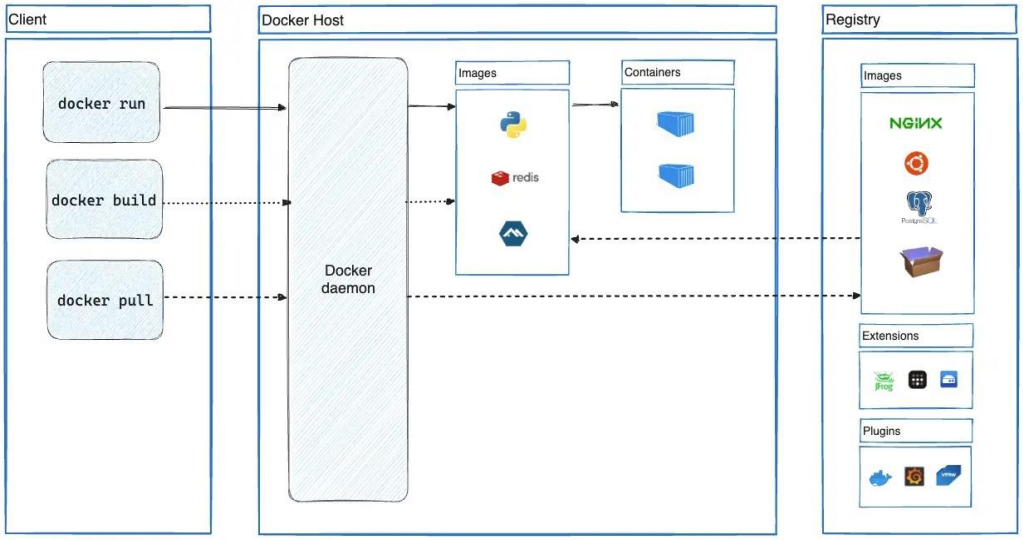
Key Features:

- Automates Workflows: Define automated tasks that execute in response to events like code pushes, pull requests, or scheduled intervals. This can include building, testing, deploying, and more.
- CI/CD Pipelines: Create workflows that chain together multiple steps, forming a CI/CD pipeline that ensures your code is built, tested, and deployed reliably.
- Community-Powered Workflows: A vast ecosystem of pre-built actions exists for various tools and services, allowing you to integrate them seamlessly into your workflows. You can also create custom actions or use containerized actions for more flexibility.
- Cross-Platform Compatibility: Actions run within Docker containers, making them compatible with any programming language and environment. You can run them on GitHub's servers or your own infrastructure.

- Flexibility: Write actions in JavaScript or use container images, allowing you to interact with the GitHub API and other public APIs. This enables customization to fit your specific needs.

2. Docker

Docker is a platform that enables developers to build, package, distribute, and run applications in containers. Containers allow you to package an application with all of its dependencies into a standardized unit, ensuring that it will run consistently across different environments.



Advantages of Using Docker:

- Consistency: Docker containers encapsulate the application, its dependencies, and the environment settings into a single unit, ensuring consistency across different development, testing, and production environments.
- Isolation: Containers provide process isolation, meaning each container operates independently of others on the same host system. This isolation ensures that changes made in one container do not affect others, improving reliability and security.
- Portability: Docker containers are lightweight and portable, allowing developers to build an application once and run it anywhere, whether on a developer's laptop, a testing environment, or a production server.

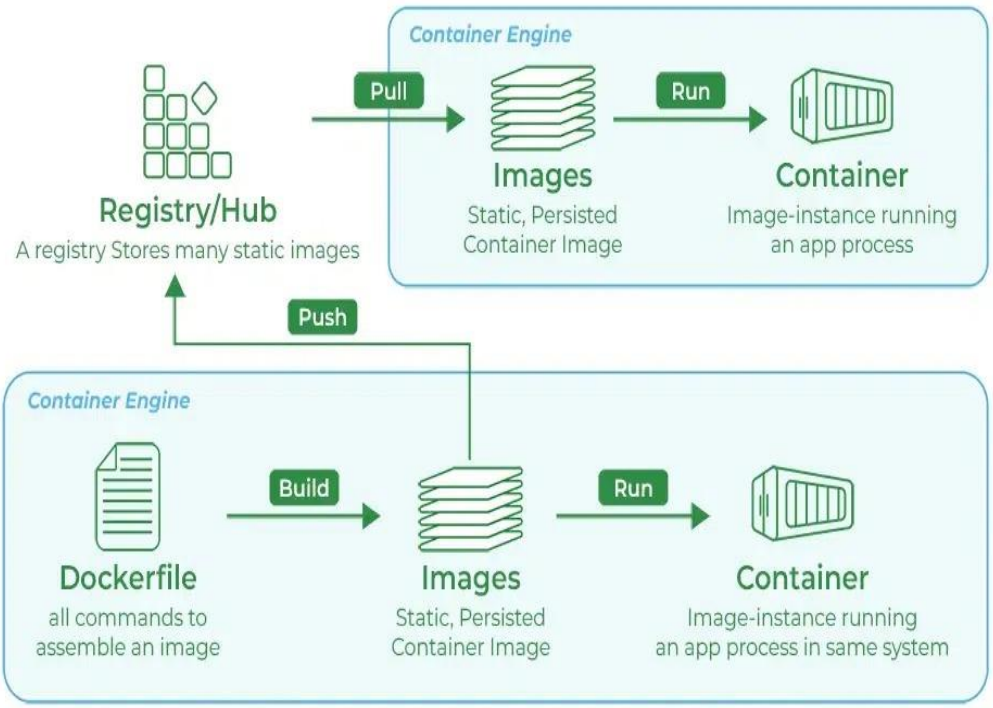
- Scalability: Docker enables easy scaling of applications by allowing you to create multiple instances of containers to handle increased load. Containers can be dynamically orchestrated using tools like Docker Swarm or Kubernetes.
- Resource Efficiency: Unlike traditional virtual machines, which require a separate operating system for each instance, Docker containers share the host system's kernel, resulting in lower overhead and better resource utilization.

3. Docker Hub

Docker Hub offers a vast library of pre-built container images for various functionalities, including machine learning frameworks like TensorFlow or PyTorch. You can leverage these pre-built images to avoid setting up the environment from scratch, saving development time. If you plan to collaborate on this project with a team, Docker Hub can be a central repository to store and share different versions of your application (containers). This allows team members to easily access and run specific versions for testing or collaboration.

1. Docker hub deployment: Create a Docker Image: First, you need to create a Docker image for your application. This involves writing a Dockerfile that specifies the configuration for your image, including base image, dependencies, and commands to run your application.
2. Build the Docker Image: Once you have the Dockerfile, you use the docker build command to build the Docker image from the Dockerfile. This will create a Docker image on your local machine.
3. Tag the Docker Image: Before pushing the Docker image to Docker Hub, you need to tag it with the appropriate repository name. The tag typically includes the Docker Hub username or organization name, repository name, and optionally a version tag.
4. Login to Docker Hub: Use the docker login command to log in to Docker Hub with your Docker Hub username and password.
5. Push the Docker Image to Docker Hub: Once logged in, you can use the docker push command to push the Docker image to Docker Hub.

6. Pulling and Running the Image: Once the image is pushed to Docker Hub, you can pull it onto any machine with Docker installed and run it as a container, as described in the previous response.
7. Managing Containers: Docker provides commands for managing containers, such as starting, stopping, restarting, and removing containers. You can also view logs and inspect container details using Docker commands.
8. Scaling and Orchestration: For production deployments, consider using Docker Swarm or Kubernetes for container orchestration. These tools provide features for deploying, managing, and scaling containers across multiple hosts, ensuring high availability and fault tolerance.



Step-by-step example:

Create a Dockerfile: Create a file named Dockerfile in your project directory with the following content:

Step 1: Containerize the Application

Create a Dockerfile (assuming your application is in a directory named 'time_management_system')

cd time_management_system

touch Dockerfile

Step 2: Build the Docker Image
docker build -t time-management-system .

3. Tag the Image:
Once the build is successful, tag the Docker image with your Docker Hub repository name

Step 3: Tag the Image
docker tag time-management-system yourusername/time-management-system:latest

4. Login to Docker Hub:
Run the following command to log in to Docker Hub: bash

Step 4: Log in to Docker Hub
docker login

5. Push the image to Docker Hub:
Finally, push the Docker image to Docker Hub:

Step 5: Push the Image to Docker Hub
docker push yourusername/time-management-system:latest

Once you have the Docker image pushed to Docker Hub, you can proceed with the deployment process. Below is a guide on how to deploy the time management system using Docker

1. Set Up Your environment: Make sure Docker is installed and running on your system. You can download and install Docker Desktop from the official Docker website if you have not already.

2. Pull the docker Image: Use the docker pull command to download the Docker image from Docker Hub to your local machine:

   docker pull yourusername/time-management-system:latest

3. Run the Docker container: Once the image is pulled, you can run a Docker container based on that image:

   docker run -d -p 8080:80 --name time-management-container yourusername/time-management-system:latest

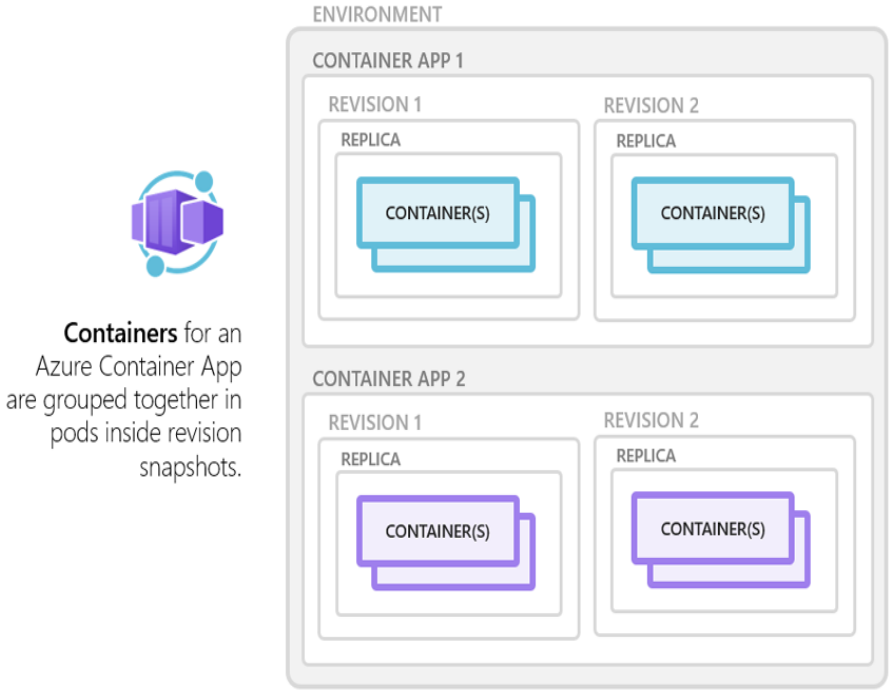
This command runs a container named Time-management-container in detached mode ('-d'), maps port 8080 on the host to port 80 on the container ('-p 8080:80') and uses the Docker image you pulled.

4. Access the Application: Once the container is running, you should be able to access the time management system through your web browser at http://localhost:8080 orhttp://<docker-machine-ip>:8080, depending on your Docker setup.
5. Managing the Container: You can manage the running container using various Docker commands. For example:
   - To stop the container: docker stop time-management-container .
   - To start the container: docker start time-management-container.
   - To remove the container: docker rm time-management-container.
   - To view logs from the container: docker logs time-management-container.

6. Scaling and Orchestration: If you need to scale the application or deploy it in a production environment, consider using Docker Swarm or Kubernetes for orchestration. These tools allow you to manage multiple containers across multiple hosts efficiently.

4. Azure Container Apps

Azure Container Apps is a fully managed serverless container service provided by Microsoft Azure. It allows developers to deploy and run containerized applications without managing the underlying infrastructure. Azure Container Apps is designed to simplify the deployment and scaling of containerized workloads, offering a serverless experience where users only pay for the resources consumed by their applications.



Key features of Azure Container Apps include:

Serverless Experience:

With Azure Container Apps, developers can focus on building and deploying their applications without worrying about managing servers or infrastructure. The service automatically provisions and scales resources based on application demand, enabling a truly serverless experience

Container Orchestration:

Azure Container Apps supports both Docker containers and Kubernetes manifests, allowing users to deploy and manage containerized applications using familiar tools and workflows. It provides integrated support for container orchestration, including automated scaling, rolling updates, and health monitoring.

Integration with Azure Services:

Azure Container Apps seamlessly integrates with other Azure services, enabling developers to leverage the full power of the Azure ecosystem. It integrates with Azure Monitor for monitoring and logging, Azure Active Directory for authentication and access control, and Azure Resource Manager for managing resources.

Automatic Scaling:

Azure Container Apps automatically scales the underlying infrastructure to match application demand, ensuring optimal performance and resource utilization. Users can define scaling policies based on metrics such as CPU usage, memory usage, or custom metrics, allowing applications to scale up or down dynamically in response to changes in workload.

High Availability:

Azure Container Apps provides built-in high availability and fault tolerance, ensuring that applications remain available and responsive even in the event of failures or outages. It automatically replicates containers across multiple availability zones and regions, minimizing downtime and ensuring continuous operation.

Pay-Per-Use Billing:

Azure Container Apps follows a pay-per-use billing model, where users only pay for the resources consumed by their applications. There are no upfront costs or long-term commitments, making it cost-effective for both small-scale and large-scale deployments.

Integrated Networking:

Azure Container Apps seamlessly integrates with Azure Virtual Network, allowing developers to deploy containerized applications securely within their virtual network environments. This enables applications to communicate with other Azure services and on-premises resources using private IP addresses.

Support for Multi-Container Applications:

Azure Container Apps supports the deployment of multi-container applications, where multiple containers are grouped together and deployed as a single unit. This enables developers to deploy complex applications composed of microservices or interconnected components.

Global Availability:

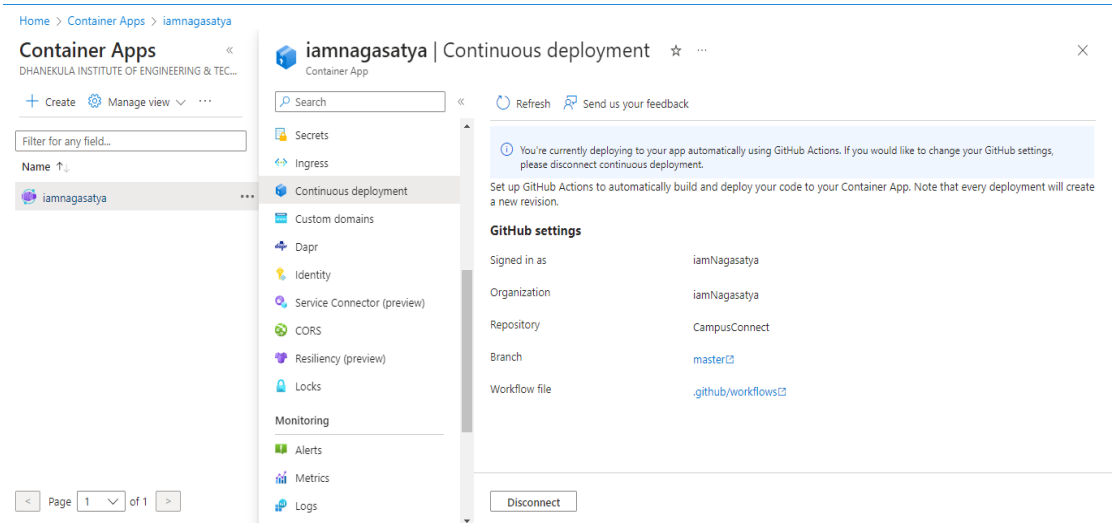
Azure Container Apps is available in multiple Azure regions worldwide, ensuring low-latency access and high availability for containerized applications. This global footprint enables developers to deploy applications closer to their users, improving performance and user experience.

Cost-Effective Pricing:

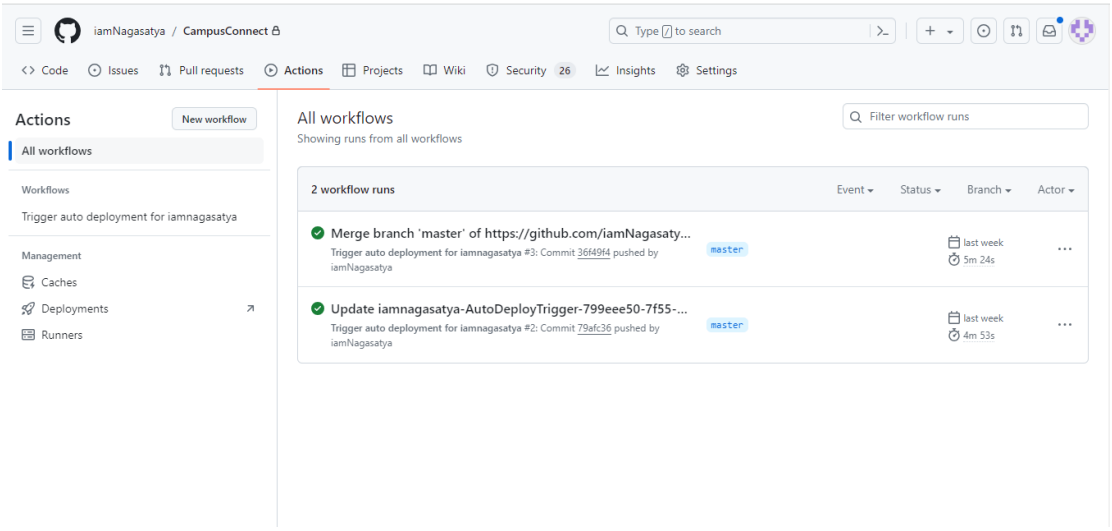
Azure Container Apps follows a pay-as-you-go pricing model, where developers only pay for the resources consumed by their applications. There are no upfront costs or long-term commitments, making it a cost-effective option for deploying containerized applications.

5. CI CD PIPELINE

Azure Container Apps seamlessly integrates with Azure DevOps, GitHub Actions, and other CI/CD tools, allowing developers to automate the deployment of containerized applications. This enables teams to achieve faster release cycles and improve overall productivity.



Azure Continuous Deployment Configuration



Azure container apps automatic deployment with GitHub Workflows

6. Azure Container registry

Azure Container Registry is a managed registry service based on the open-source Docker Registry 2.0. Create and maintain Azure container registries to store and manage your container images and related artifacts. Use Azure container registries with your existing container development and deployment pipelines, or use Azure Container Registry Tasks to build container images in Azure. Build on demand, or fully automate builds with triggers such as source code commits and base image updates.

Use cases

- Pull images from an Azure container registry to various deployment targets
- Scalable orchestration systems that manage containerized applications across clusters of hosts, including Kubernetes, DC/OS, and Docker Swarm.

Azure services that support building and running applications at scale, including Azure Kubernetes Service (AKS), App Service, Batch, Service Fabric, and others.

Developers can also push to a container registry as part of a container development workflow. For example, target a container registry from a continuous integration and delivery tool such as Azure Pipelines or Jenkins.

Configure ACR Tasks to automatically rebuild application images when their base images are updated, or automate image builds when your team commits code to a Git repository. Create multi-step tasks to automate building, testing, and patching multiple container images in parallel in the cloud.

Azure provides tooling including the Azure CLI, the Azure portal, and API support to manage your Azure container registries. Optionally install the Docker Extension for Visual Studio Code and the Azure Account extension to work with your Azure container registries. Pull and push images to an Azure container registry, or run ACR Tasks, all within Visual Studio Code.

Key features

- **Registry service tiers** - Create one or more container registries in your Azure subscription. Registries are available in three tiers: Basic, Standard, and Premium, each of which supports webhook integration, registry authentication with Microsoft Entra ID, and delete functionality. Take advantage of local, network-close storage of your container images by creating a registry in the same Azure location as your deployments. Use the geo-replication feature of Premium registries for advanced replication and container image distribution scenarios.

- **Security and access** - You log in to a registry using the Azure CLI or the standard docker login command. Azure Container Registry transfers container images over HTTPS, and supports TLS to secure client connections.

  You control access to a container registry using an Azure identity, a Microsoft Entra ID-backed service principal, or a provided admin account. Use Azure role-based access control (Azure RBAC) to assign users or systems fine-grained permissions to a registry.

  Security features of the Premium service tier include content trust for image tag signing, and firewalls and virtual networks (preview) to restrict access to the registry. Microsoft Defender for Cloud optionally integrates with Azure Container Registry to scan images whenever an image is pushed to a registry.

- **Supported images and artifacts** - Grouped in a repository, each image is a read-only snapshot of a Docker-compatible container. Azure container registries can include both Windows and Linux images. You control image names for all your container deployments. Use standard Docker commands to push images into a repository, or pull an image from a repository. In addition to Docker container images, Azure Container Registry stores related content formats such as Helm charts and images built to the Open Container Initiative (OCI) Image Format Specification.
- **Automated image builds** - Use Azure Container Registry Tasks (ACR Tasks) to streamline building, testing, pushing, and deploying images in Azure. For example, use ACR Tasks to extend your development inner-loop to the cloud by offloading docker build operations to Azure. Configure build tasks to automate your container OS and framework patching pipeline, and build images automatically when your team commits code to source control.

  Multi-step tasks provide step-based task definition and execution for building, testing, and patching container images in the cloud. Task steps define individual container image build and push operations. They can also define the execution of one or more containers, with each step using the container as its execution environment.

7. Azure Database for PostgreSQL

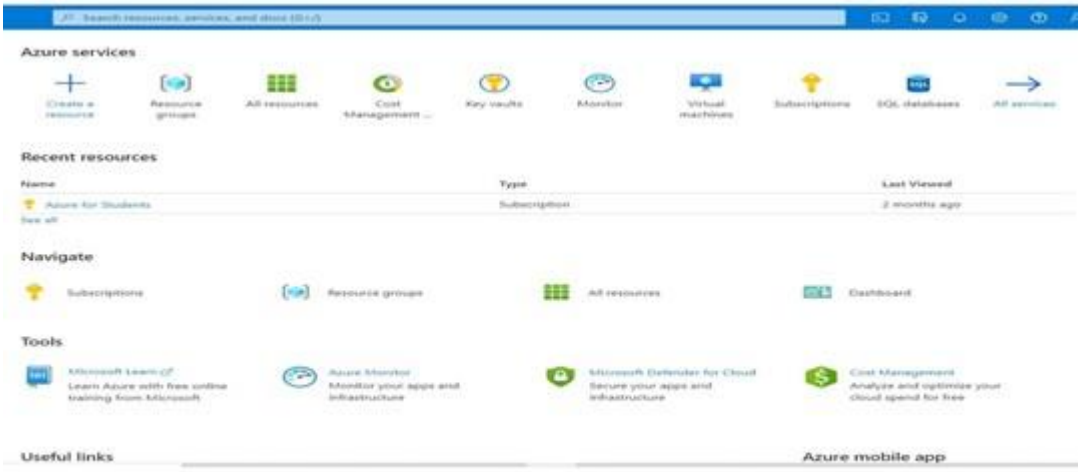
Azure provides us with a platform-as-a-service implementation of PostgreSQL in Azure with an Azure Database for PostgreSQL. This service provides us with the same administrative benefits, scalability, security, performance, and availability as the MySQL service.

Azure Database for PostgreSQL has 3 different deployment options that we may use:

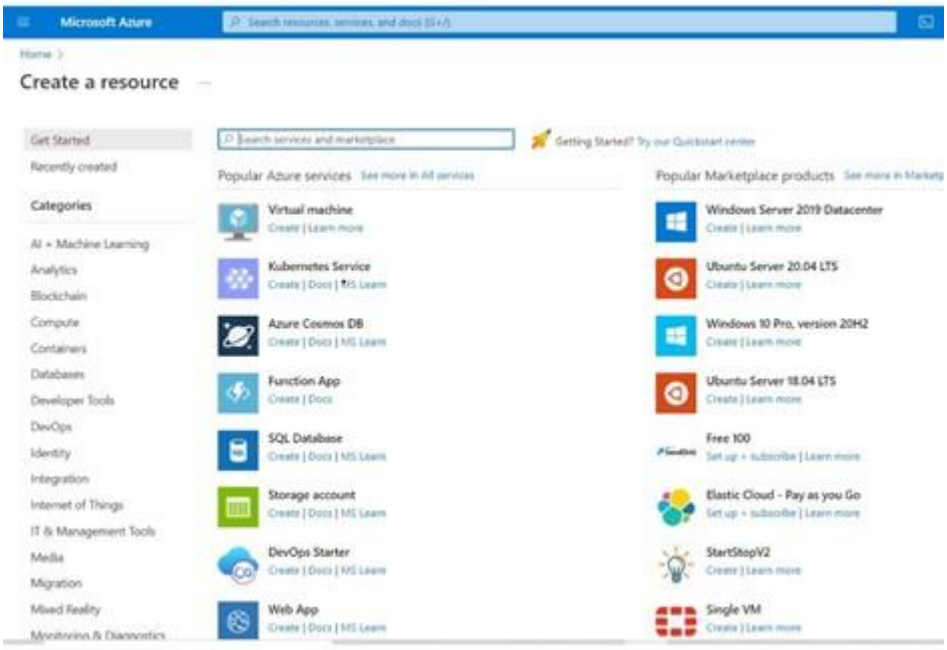
- **Single server**: This deployment option for the Azure Database for PostgreSQL provides us with similar benefits as the Azure Database for MySQL. We may choose from the three pricing tiers: Basic, General Purpose, and memory-optimized based on our needs (the load which we are required to support). Each of the tiers supports different memory, storage size, and number of CPUs.
- **Flexible server**: The flexible server deployment option for the Azure Database for PostgreSQL is a fully managed database service by Azure. It provides us with more control, server configuration customizations, and better cost optimization controls.
- **Hyperscale**: The Hyperscale Citus is a deployment option available for the Azure Database for PostgreSQL that supports large database loads by scaling queries across multiple server nodes. The database is split across nodes and the data is split into chunks based on the value of sharing or partition key. We should use this deployment option when there are large database PostgreSQL deployments.

Create an Azure Database for PostgreSQL

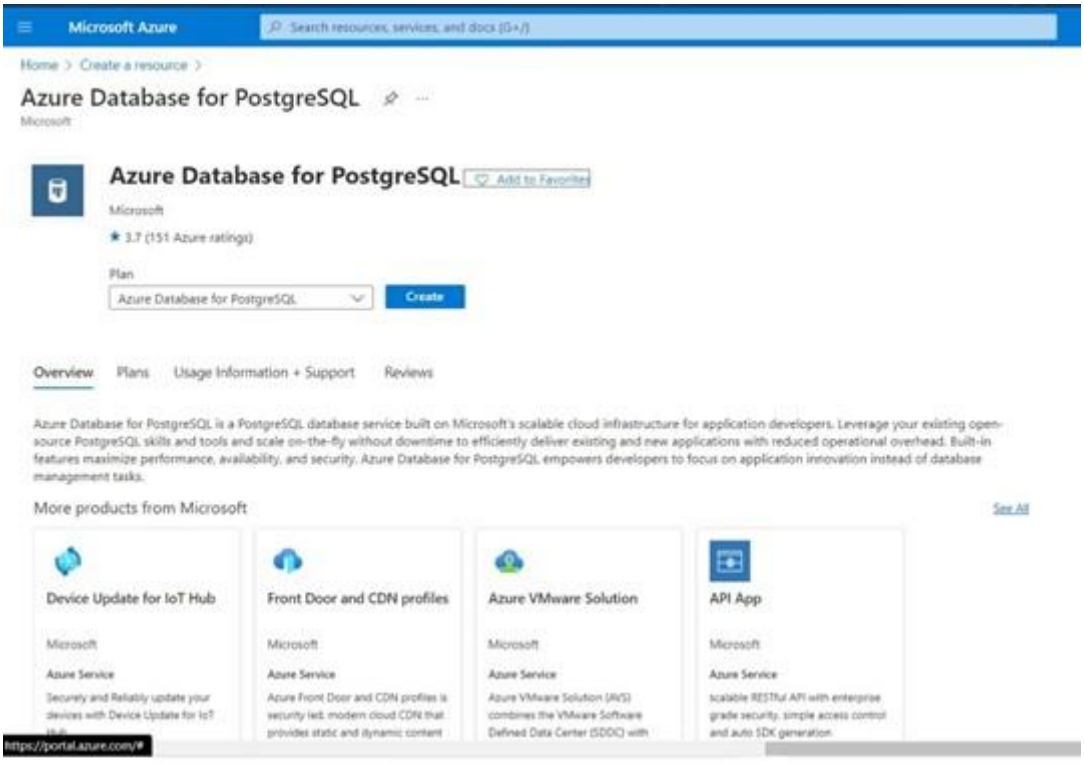
Step 1: Navigate to the Azure portal



Step 2: In the Azure Portal, select the + Create a resource option. You will see a list of resources we can provide in Azure.

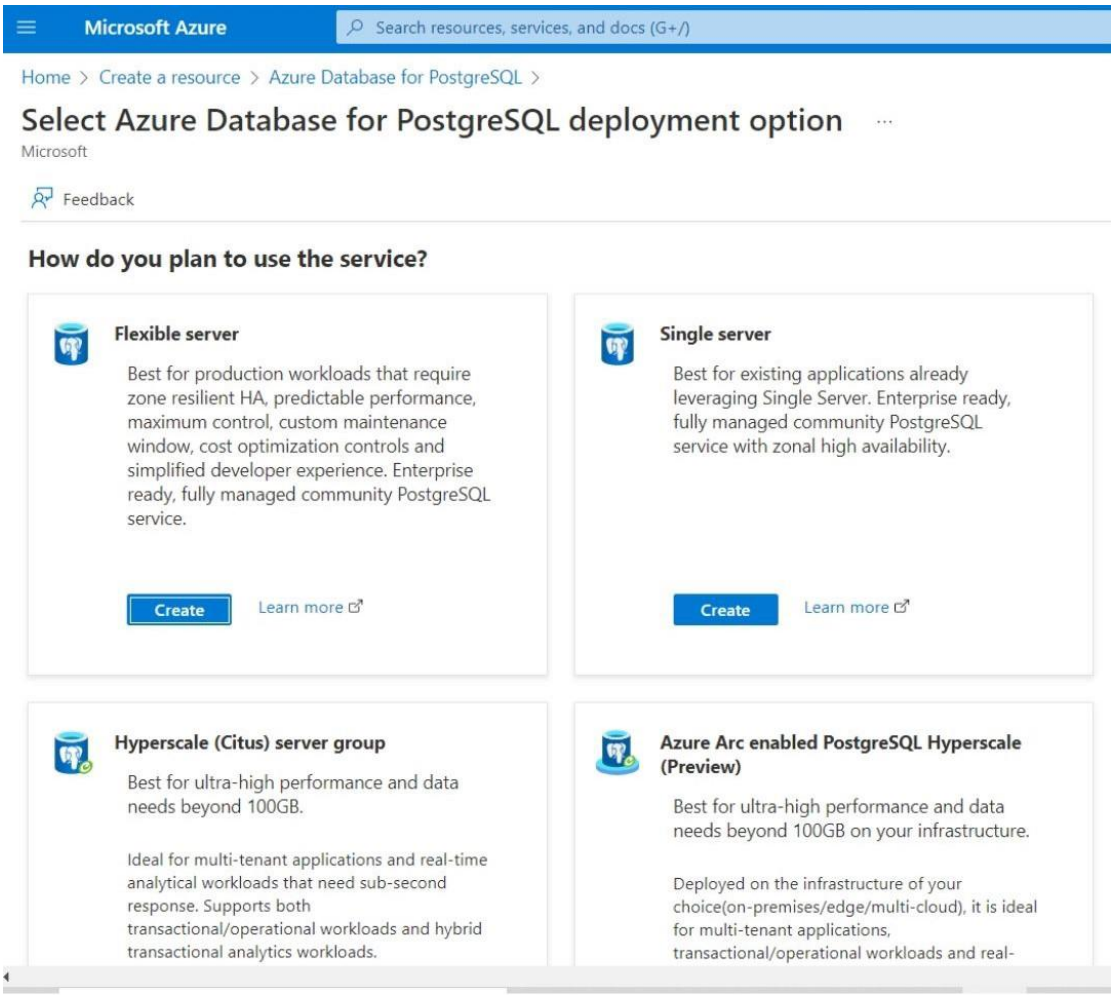


Step 3: In the search type Azure Database for PostgreSQL.



Step 4: In the resulting page select create option.

Step 5: Review the different options available for the Azure Database for PostgreSQL. Choose the one that you need as per your requirements. For now, select the single server tile and select create. Decline the offer to switch to a flexible server if asked.

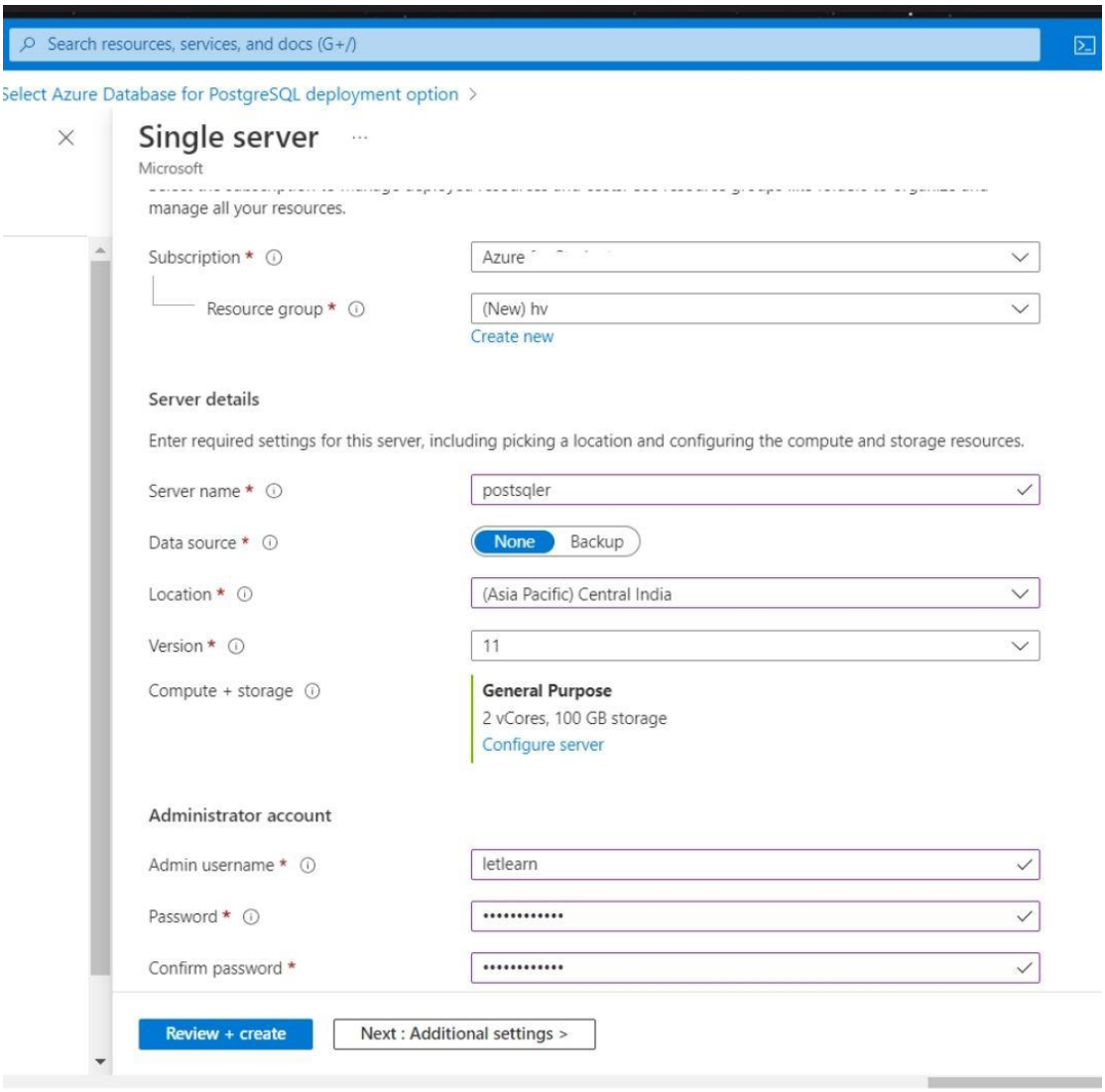


Step 6: You will be prompted with a create SQL database page.

Enter the details:

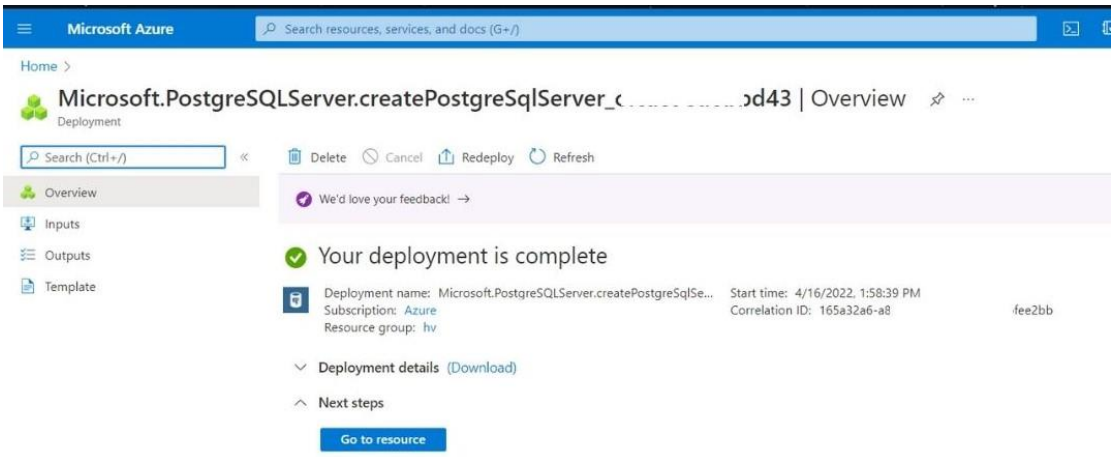
- Subscription: Select the Azure subscription where you wish to create the resource.
- Resource group: Choose a resource group where you wish to create the resource. If you wish to create a new one click on create a new option.
- Server name: Give a unique name to your server.
- Data source: Choose none.

- Location: Choose a location from the list.
- Version: Leave it to the default value.
- Compute + Storage: For now, select configure server option and then change vCore to two cores. Select OK.
- Admin username: Give a username.
- Password: Give a complex password for your database.
- Confirm password: Retype the password for confirmation.

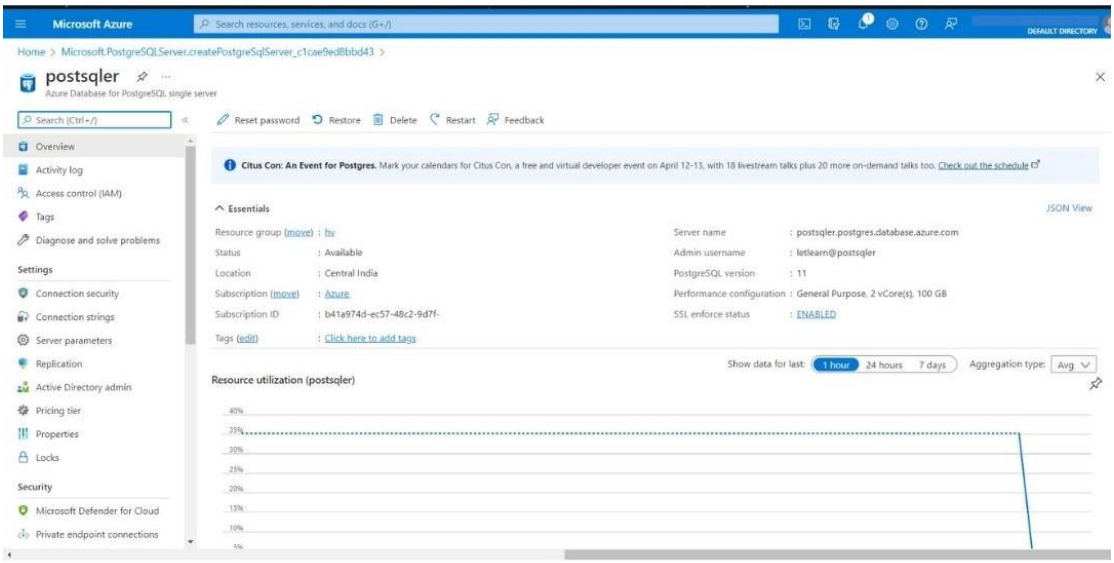


Step 7: Select the Review + Create option.

Step 8: After reviewing all the details click on create to create your Azure Database for PostgreSQL.



Step 9: Wait for the deployment to complete. Then click on go to resource option to go to your deployed resource.



Benefits of Azure Database for PostgreSQL:

- It contains built-in failure detection and failover mechanisms which makes it a highly available service.
- We can use the pgAdmin tool to connect to the Azure Database for PostgreSQL to manage and monitor it. However, some of the server-focused functionalities such as server backup and restore aren't available.

- It records the information about queries run on the database in the server which is saved in a database name azue_sys. We can query the query_store.qs_view to get the information and use it to monitor the queries the users are running.

8. Custom Domain Management

Azure Container Apps allows you to bind one or more custom domains to a container app. You can automatically configure a free managed certificate for your custom domain.

Free certificate requirements

Azure Container Apps provides a free managed certificate for your custom domain. Without any action required from you, this TLS/SSL server certificate is automatically renewed as long as your app continues to meet the requirements for managed certificates.

The requirements are:

- Your container app has HTTP ingress enabled and is publicly accessible.
- For apex domains, you must have an A record pointing to your Container Apps environment's IP address.

For subdomains, you must have a CNAME record mapped directly to the container app's automatically generated domain name. Mapping to an intermediate CNAME value blocks certificate issuance and renewal. Examples of CNAME values are traffic managers, Cloudflare, and similar services.

Add a custom domain and managed certificate

- Navigate to your container app in the Azure portal
- Verify that your app has HTTP ingress enabled by selecting Ingress in the Settings section. If ingress isn't enabled, enable it with these steps:
    - Set HTTP Ingress to Enabled.
    - Select the desired Ingress traffic setting.

    - Enter the Target port.
    - Select Save.
- Under the Settings section, select Custom domains.
- Select Add custom domain.
- In the Add custom domain and certificate window, in TLS/SSL certificate, select Managed certificate.
- In domain, enter the domain you want to add.
- Select the Hostname record type based on the type of your domain.
- Using the DNS provider that is hosting your domain, create DNS records based on the Hostname record type you selected using the values shown in the Domain validation section. The records point the domain to your container app and verify that you are the owner
- Select Validate.
- Once validation succeeds, select Add.
    - It may take several minutes to issue the certificate and add the domain to your container app.
- Once the operation is complete, you see your domain name in the list of custom domains with a status of Secured. Navigate to your domain to verify that it's accessible.

RESULTS

7. RESULTS

OUTPUT SCREENS

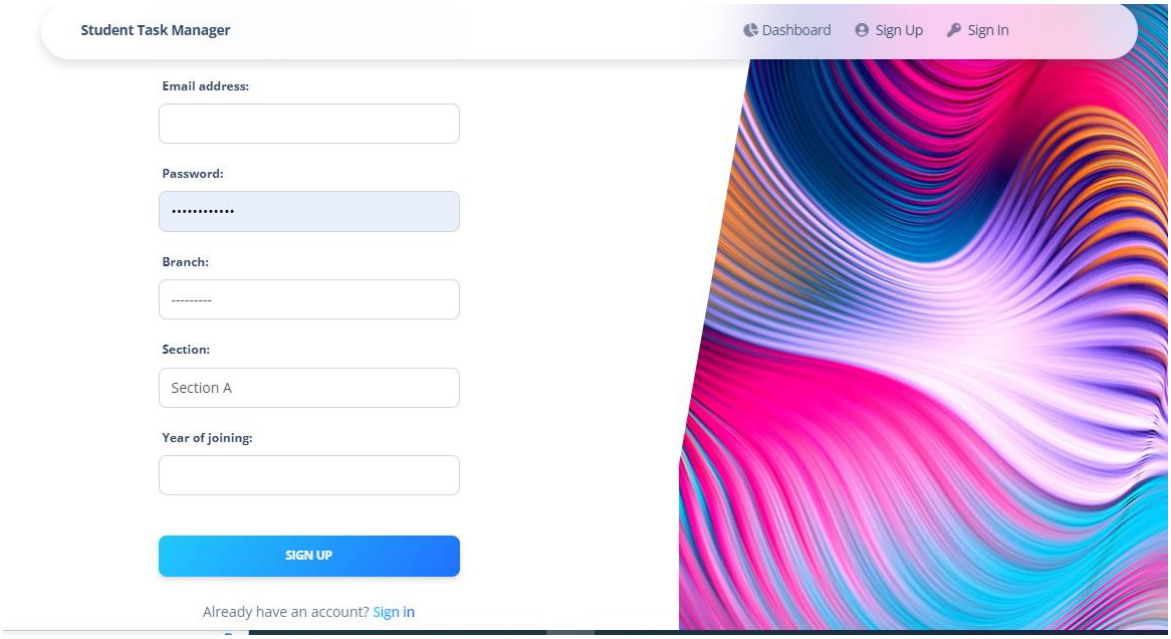


Figure 7.1: Sign-Up Page

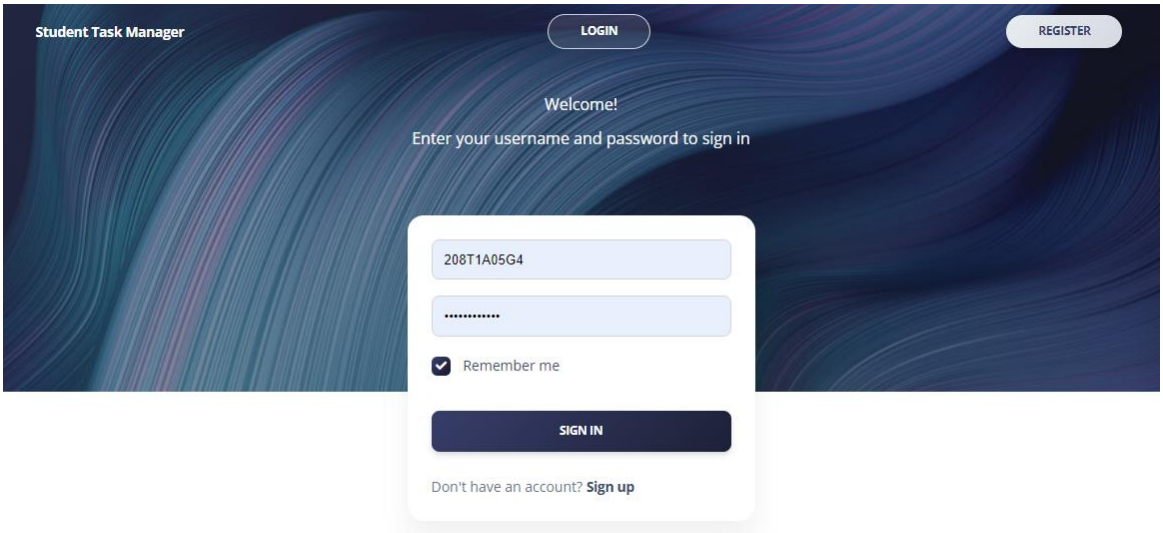


Figure 7.2: Login Page

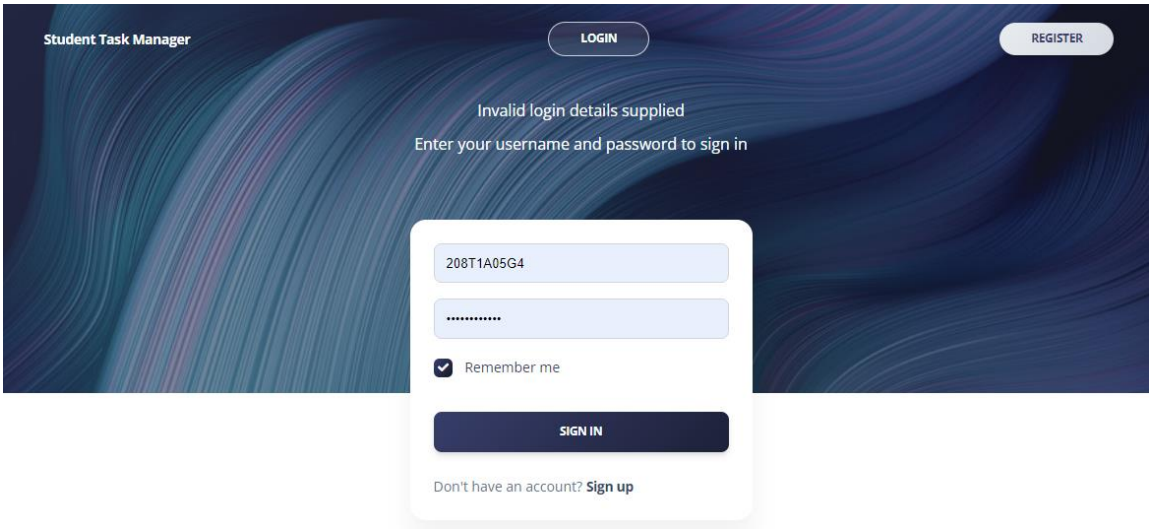


Figure 7.3: Invalid login

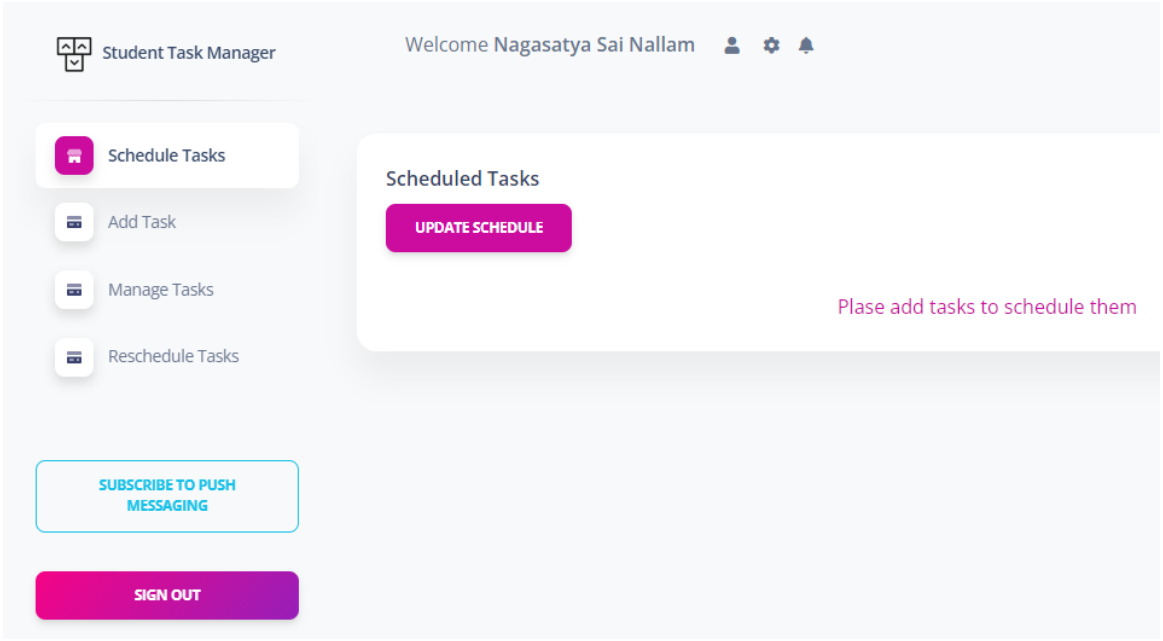


Figure 7.4: Successful login

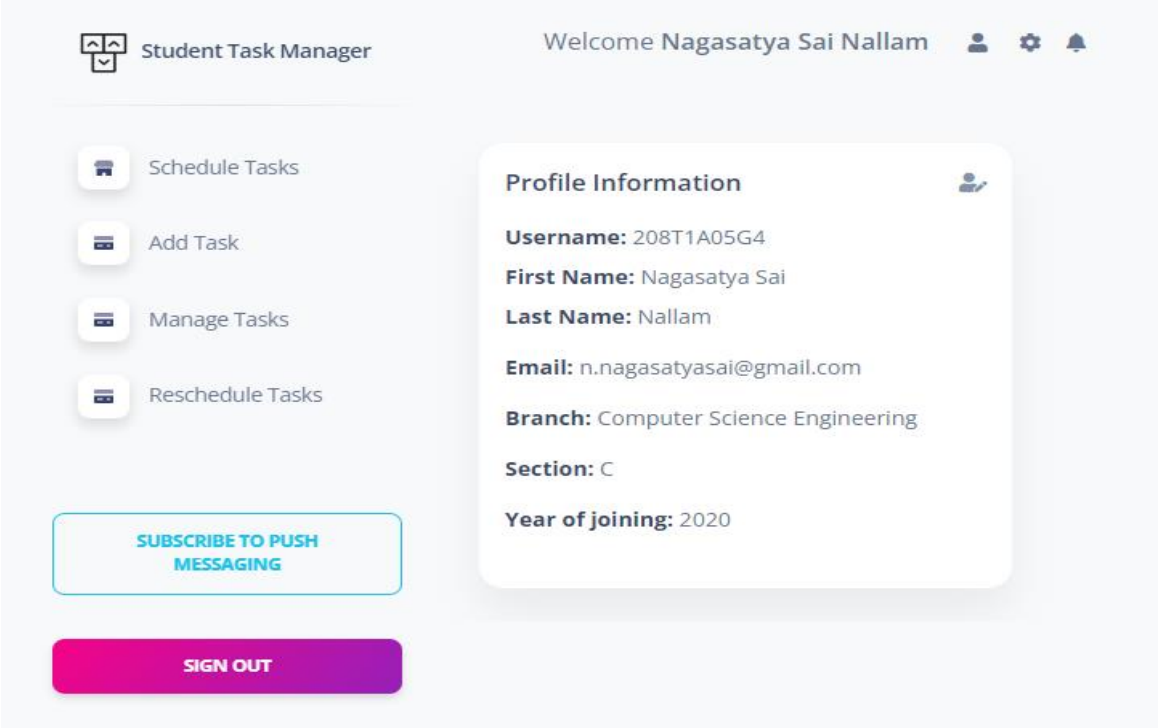


Figure 7.5: Student Profile details

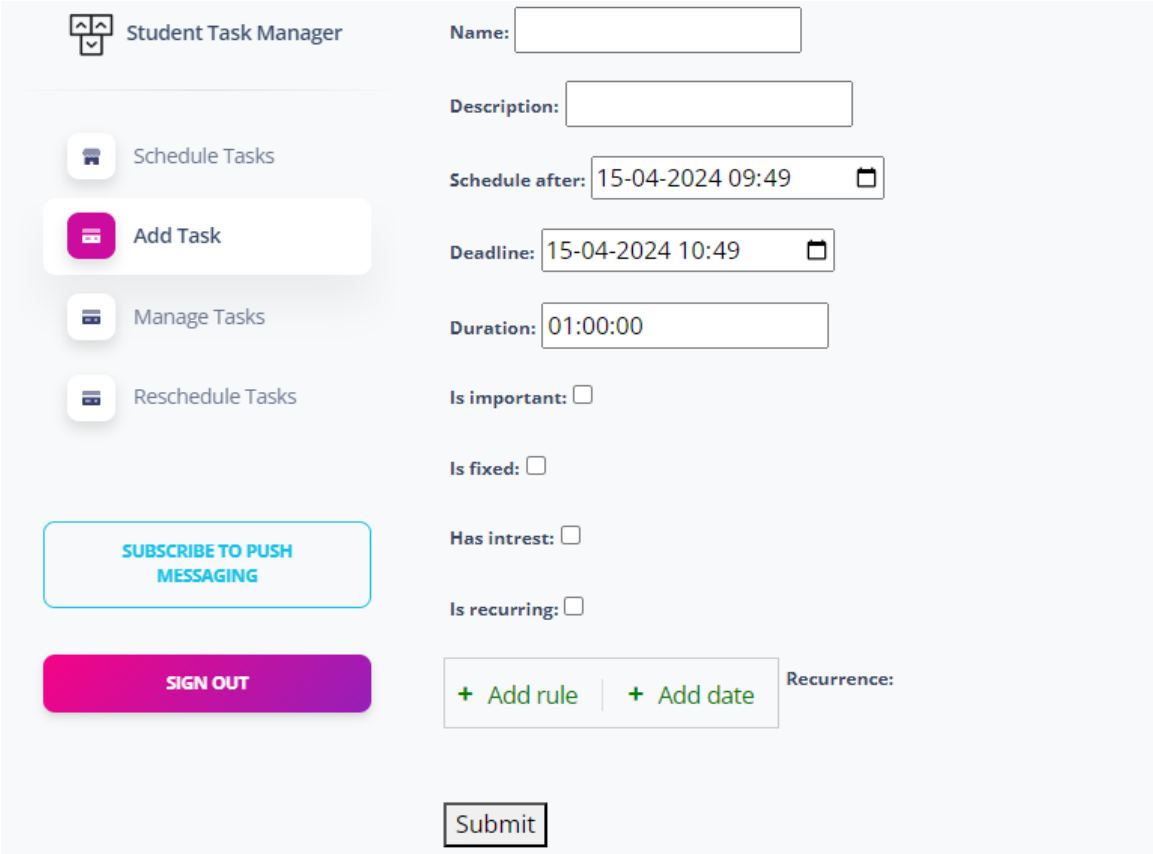


Figure 7.6: Add task

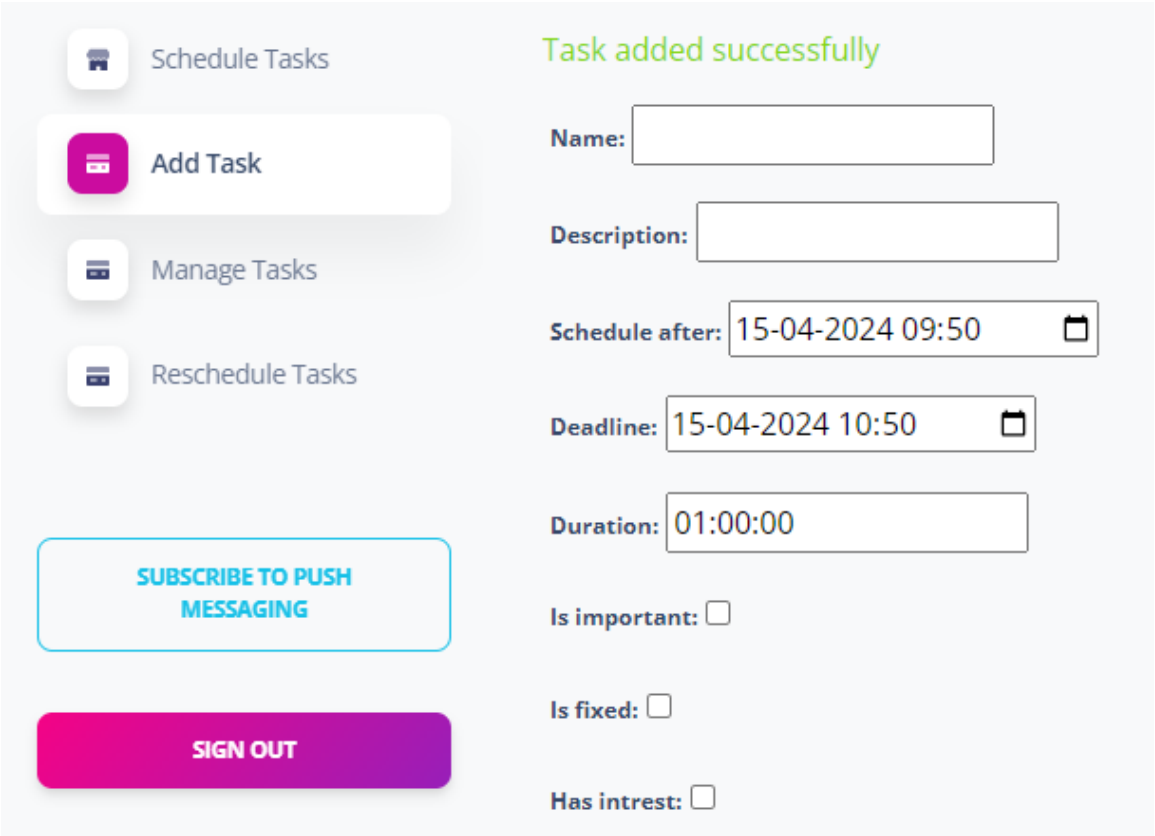


Figure 7.7: Task added successfully

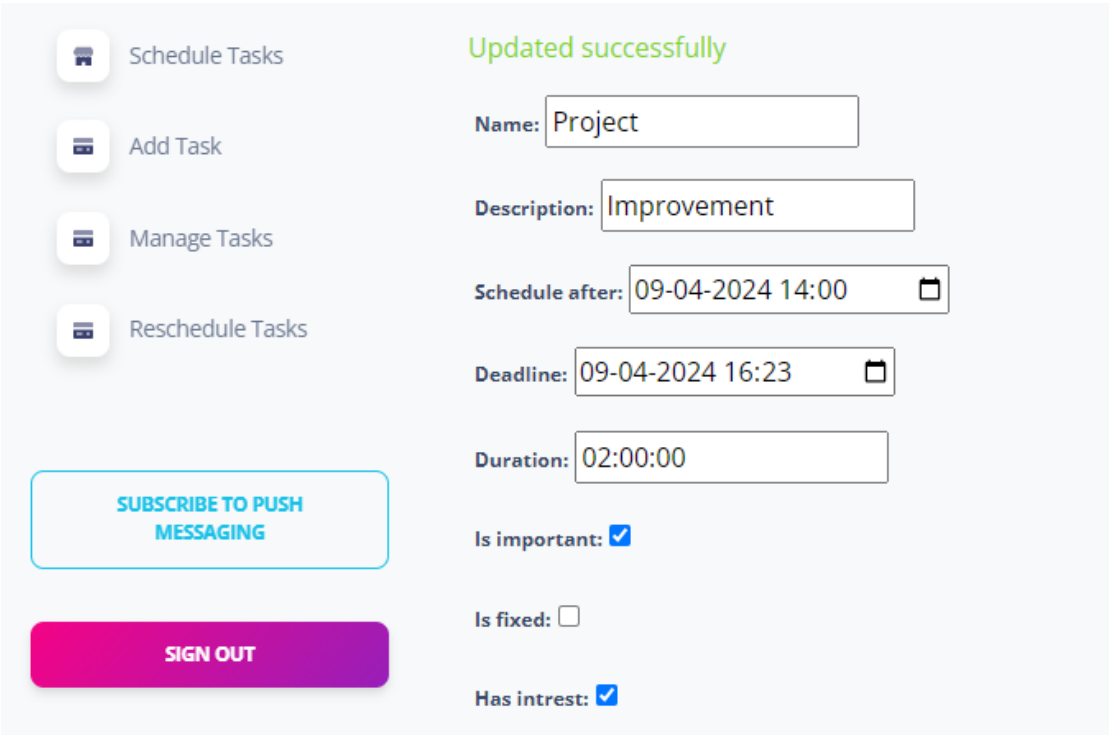


Figure 7.8: Task Updated Successfully

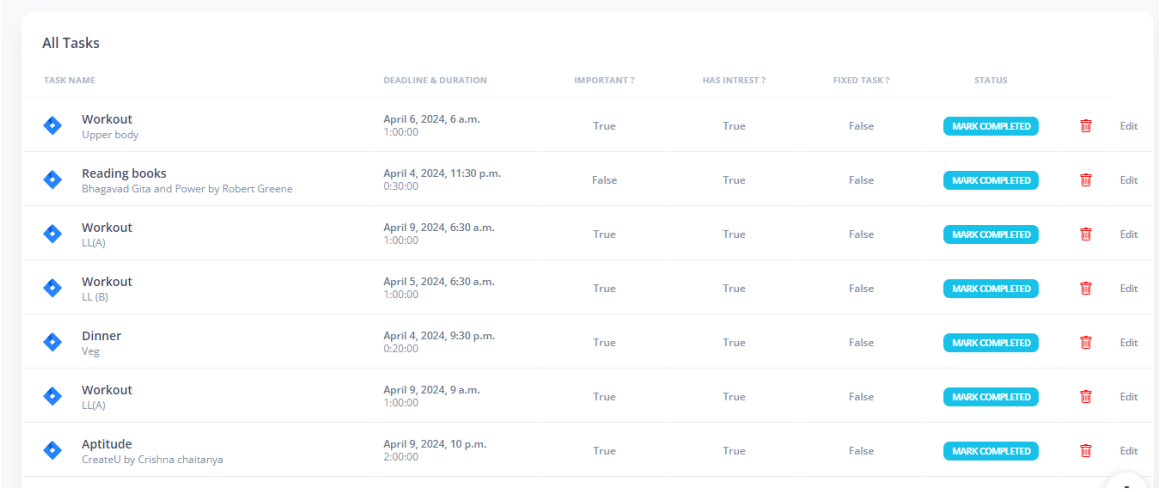


Figure 7.9: Manage Tasks (mark status & deletion)

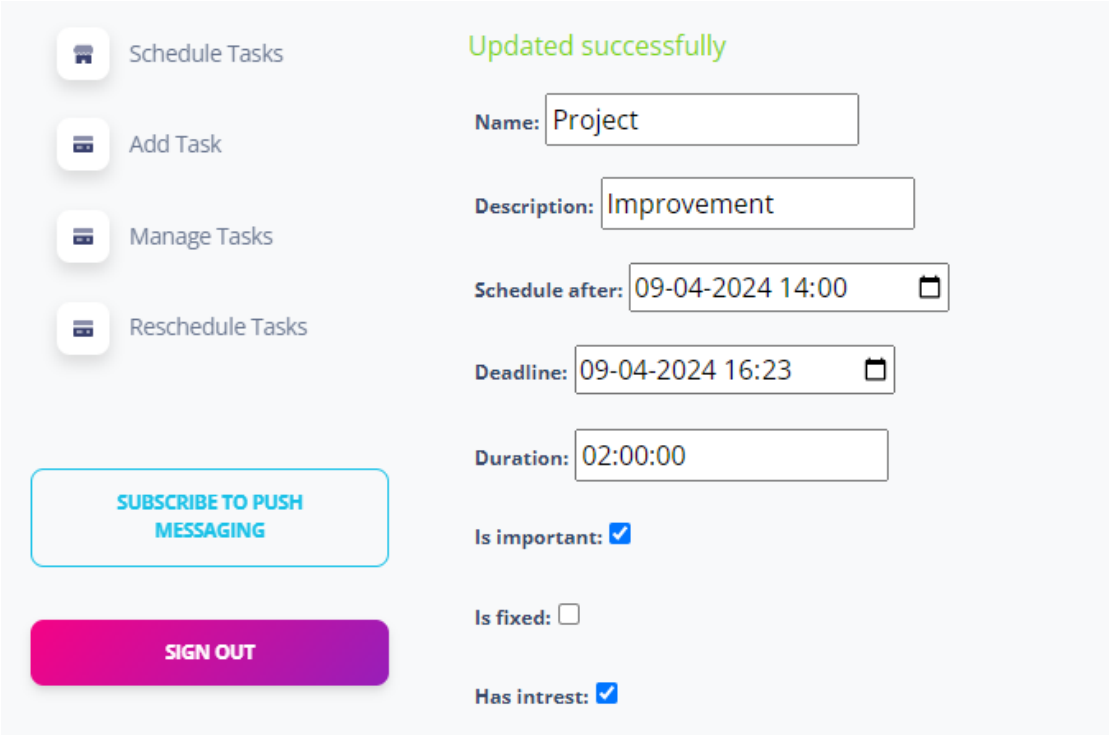


Figure 7.10: Task Updated Successfully

Figure 7.11: Schedule Tasks

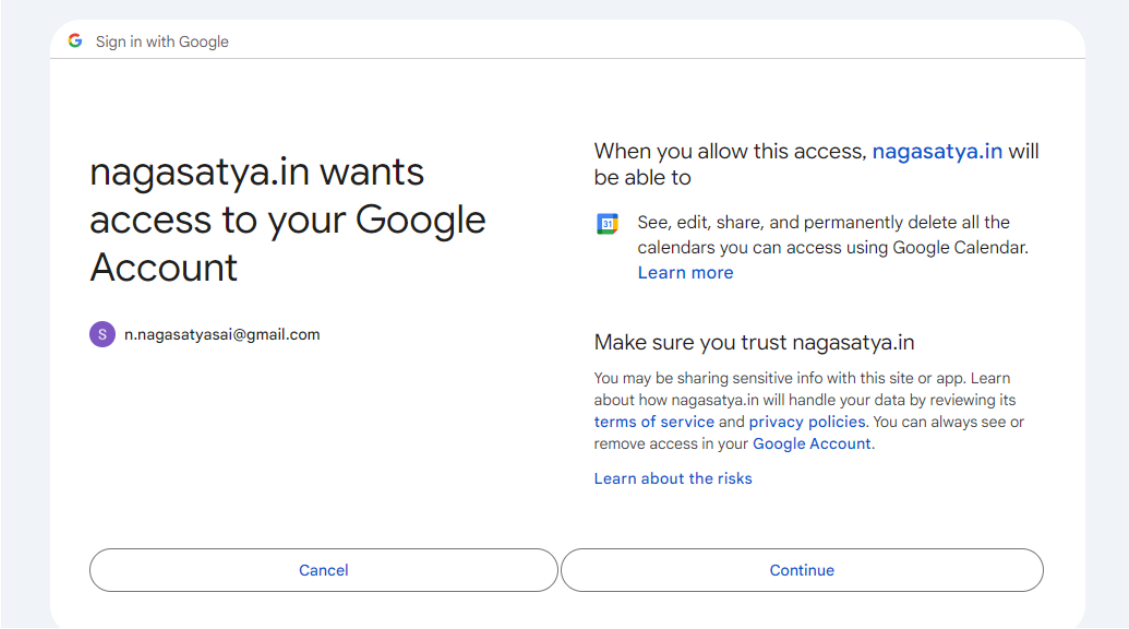


Figure 7.12: Google Calendar Integration

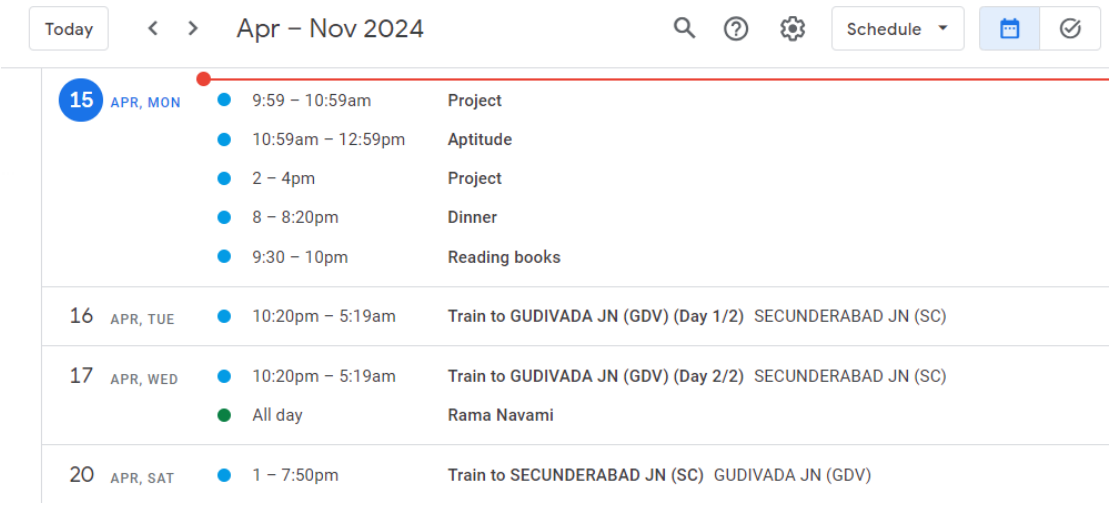


Figure 7.13: Synchronising Scheduled tasks to Google Calendar

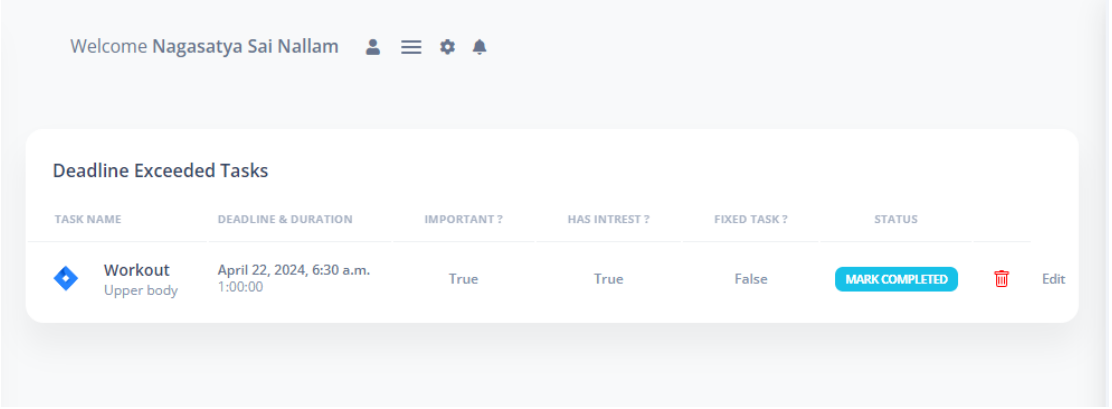


Figure 7.14: Rescheduling tasks

CONCLUSION

7. CONCLUSION

In conclusion, this project presents a groundbreaking approach to addressing the critical challenge of time management in the digital era. Leveraging Actor-Critic Reinforcement learning, it pioneers a dynamic and personalized system for optimizing productivity and effectiveness. By embracing adaptive scheduling and individualized strategies, this innovative solution aims to empower users across various domains to thrive in today's rapidly evolving digital landscape. Through the utilization of Reinforcement Learning, this solution provides individualized techniques to optimize productivity by adapting to the individual demands of users. By employing dynamic learning and adaptation, this system surpasses the constraints of static scheduling models, providing a versatile and effective method for task planning. In essence, our goal is to provide individuals with the necessary tools to excel in the modern, rapidly changing digital landscape, hence promoting efficiency and achievement in various industries.

FUTURE SCOPE

8. FUTURE SCOPE

The future scope for the project outlined in the abstract is promising and multifaceted. Here are some potential directions for further development and exploration. Continuously refine and optimize the reinforcement learning algorithms to enhance the system's adaptability and effectiveness in addressing individual preferences and behaviors. Integrate the intelligent time management system with existing AI assistants or productivity tools to provide seamless and intuitive user experiences. Extend the application of the system beyond the academic realm to cater to professionals, freelancers, and individuals in various industries, offering tailored time management solutions for diverse contexts. Incorporate long-term planning and goal-setting capabilities into the system, allowing users to set and track their objectives over extended time horizons. Utilize advanced behavioral analytics to gain insights into user habits, preferences, and productivity patterns, enabling further customization and refinement of the time management strategies. Explore integration with mobile devices and wearable technologies to provide real-time notifications, reminders, and feedback on time management activities. Develop collaborative features that enable teams or groups to synchronize schedules, coordinate tasks, and optimize collective productivity. Conduct research on the ethical implications of implementing AI-driven time management systems, addressing concerns related to privacy, autonomy, and psychological well-being.

Overall, the future scope involves continuous innovation and interdisciplinary collaboration to further enhance the capabilities and impact of the proposed intelligent time management system.

REFERENCE

9. REFERENCE

[1]“Automated Time Manager: Effectiveness of Self-Regulation on Time Management through a Smartphone Application” 10.1109/ACCESS.2019.2926743, IEEE Access.

[2] “ScheduleME - Smart Digital Personal Assistant for Automatic Priority Based Task Scheduling and Time Management” 2021 2nd Global Conference for Advancement in Technology (GCAT) Bangalore, India. Oct 1-3, 2021.

[3] R. V. Adams and E. Blair, "Impact of Time Management Behaviors on Undergraduate Engineering Students’ Performance - Richelle V. Adams, Erik Blair, 2019", SAGE Journals, 2019.

[4] “Developing Real-Time Scheduling Policy by Deep Reinforcement Learning” 2021 IEEE 27th Real-Time and Embedded Technology and Applications.

[5] Nominal Batbaatar, Grace Amin (2021), “Students’ Time Management During Online Class”, Research Gate, pp 189.

[6] Mind Tools Content Team, “What Is Time Management? Working Smarter to Enhance Productivity”, accessed on November 22, 2021 at 02.47.

[7] J. Anderson and A. Srinivasan, “Early-release fair scheduling,” in Proceedings 12th Euromicro Conference on Real-Time Systems. Euromicro RTS.

[8] Ravichandran S. (2018) “Hands-on Reinforcement Learning with Python: Master Reinforcement and Deep Reinforcement Learning Using OpenAI Gym and TensorFlow”, Packt Publishing Ltd.

[9] Scott Fujimoto, Herke van Hoof, David Meger (2018) “Addressing Function Approximation Error in Actor-Critic Methods” Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

[10] Zhiping Peng, Delong Cui, Jinglong Zuo, Qirui Li, Bo Xu, Weiwei Lin (2015) “Random task scheduling scheme based on reinforcement learning in cloud computing” Cluster Comput (2015) 18:1595–1607DOI 10.1007/s10586-015-0484-2.

[11] Django Documentation - https://docs.djangoproject.com/en/5.0/

[12] Django Recurrence Docs https://django-recurrence.readthedocs.io/en/latest/

PUBLISHED PAPER

WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Efficient Time Management in the Digital Era: Reinforcement Learning in Action

G. Subhashini [a], N. Nagasatya Sai [b], T. Srihari [b], K. Bindu Sri [b], L. Manogna Swetha [b], D. Lakshmi Sasi Rekha [b]

[a] *Assistant Professor, Dept of Computer Science & Engineering, Dhanekula Institute of Engineering & Technology, A.P, India*
[b] *Research Scholar, Dept of Computer Science & Engineering, Dhanekula Institute of Engineering & Technology, A.P, India*

A B S T R A C T

In the contemporary digital landscape, effective time management is paramount for individuals across various domains. This paper explores a novel approach to tackling time management problems through Actor-Critic Reinforcement learning. Reinforcement learning, a powerful machine learning paradigm, is the core methodology. It serves as the foundation for developing an intelligent time management system that adapts to the unique behaviors and preferences of individuals. This system provides personalized time management strategies, enhancing productivity and overall effectiveness. Unlike conventional methods, which may involve static scheduling models, this paper embraces dynamic learning and adaptation through reinforcement learning algorithms. By doing so, it offers a flexible solution for the students to optimally manage their time via efficient planning of tasks. In summary, the project seeks to address the universal issue of efficient time management in the digital age by harnessing the capabilities of reinforcement learning.

Keywords: Time management, Actor-Critic Reinforcement Learning, Intelligent task scheduling, Dynamic Scheduling, Task Scheduling

I. INTRODUCTION

In the fast-evolving digital landscape of today, the ability to manage time efficiently stands as a cornerstone skill for individuals navigating diverse professional and personal spheres. This paper undertakes a comprehensive exploration of an innovative approach aimed at confronting the multifaceted challenges of time management, employing Actor-Critic Reinforcement Learning as its central framework. Reinforcement learning, a sophisticated machine learning paradigm renowned for its adaptability and robustness, serves as the linchpin of this endeavour, offering a powerful methodology to construct an intelligent time management system tailored to individual needs. At the heart of this system lies the concept of personalized time management strategies, meticulously crafted to accommodate the nuanced behaviours and preferences of each user.

Unlike traditional approaches characterized by rigid, one-size-fits-all scheduling models, the methodology advocated in this paper prioritizes dynamic learning and adaptation facilitated by reinforcement learning algorithms. By leveraging this dynamic framework, the system endeavours to furnish users, particularly students, with the tools necessary to optimize their time allocation and task prioritization, thereby fostering enhanced productivity and efficacy in their endeavours. A departure from conventional wisdom, which often espouses static planning paradigms, this paper champions a paradigm shift towards agility and responsiveness in time management practices. Through its innovative utilization of reinforcement learning, the proposed system offers a malleable and adaptable solution capable of accommodating the fluid nature of modern-day commitments and obligations. In essence, this research initiative represents a concerted effort to address the pervasive challenge of effective time management in the digital era by harnessing the transformative potential of advanced machine learning techniques.

II. LITERATURE SURVEY

"***Effectiveness of Self-Regulation on Time Management through a Smartphone Application***" by Bogoan Kimi and Huajong Hong investigated the effectiveness of a self-regulation strategy on time management leveraged by smartphone capabilities using a theoretical framework of self-regulation that consists of goal setting, task strategy utilization, self-monitoring, and reflection.

"***Application of Software Engineering in Student Time Management using Prototype Mode***" by Exaudina Glory Sianturi, Sharon Cedila Suryadi constructed and develop a scheduler application, which uses Prototyping Model as the development method and traditional scheduling algorithms for creating the schedule.

"***Developing Real-Time Scheduling Policy by Deep Reinforcement Learning***" by Zitong Bo and Ying Qiao, designed a pattern to make the multiprocessor scheduling policies perform well under various task loads. In this paper, they investigated a new real-time scheduling policy based on reinforcement learning.

"***Personalized Time Management Systems: A Review***" by John Doe et al.: This paper reviews existing time management systems, highlighting their limitations in adapting to individual preferences and behaviors, thus motivating the need for personalized approaches like the one proposed.

"***Adaptive Learning Systems for Task Planning***" by Alice Johnson et al.: This study explores adaptive learning systems for task planning, drawing parallels with the proposed intelligent time management system that dynamically adjusts strategies based on user feedback and behavior.

III. EXISTING SYSTEM

- Existing systems use the traditional scheduling algorithms like Branch and Bound and Earliest release time algorithms to schedule the given set of tasks.
- Traditional schedulers typically suffer from one of two drawbacks: high computational load or poor performance. New algorithms that learn from related problems may generalize well, finding near-optimal schedules in a shorter, more practical amount of runtime.
- Also, there is no option to give priorities to the tasks like importance, user preference and to schedule the tasks upon the priorities.

IV. PROPOSED SYSTEM

- The Proposed System collects and stores all the tasks the student must do, prioritize them according to their importance, schedule them intelligently across the student's remaining time considering his/her existing academic and personal timetables and daily routines.
- A user-friendly and comprehensible web application is designed where the right amount of information is presented to the user that user can configure and override.
- Unlike the existing system, the proposed system uses the Actor Critic Reinforcement learning algorithm to optimally schedule the tasks with minimum runtime bottleneck.

V. TASK SCHEDULING - TRADITIONAL SCHEDULERS

Task scheduling is a fundamental problem in computer science and operations research, aiming to allocate resources and determine the execution sequence of tasks to optimize various performance metrics such as make span, resource utilization, and throughput. In this section, we delve into the basics of task scheduling, exploring different scheduling algorithms including Branch and Bound optimal, Random Scheduling, and Earliest Release Time (ERT).

1. Branch and Bound Optimal:

Branch and Bound is a classic algorithmic technique used to solve combinatorial optimization problems such as task scheduling. The essence of Branch and Bound lies in systematically exploring the solution space by branching into subproblems and bounding the search based on certain criteria. In the context of task scheduling, Branch and Bound aims to find the optimal schedule by recursively partitioning the set of feasible schedules and pruning branches that cannot lead to better solutions than the current best known solution.

The Branch and Bound approach starts with an initial solution and iteratively explores different scheduling possibilities, branching into subproblems to exhaustively search for the optimal schedule. At each step, the algorithm evaluates the feasibility and optimality of potential schedules, updating the best known solution based on specific criteria such as minimizing makespan or maximizing resource utilization.

Despite its effectiveness in finding optimal solutions, Branch and Bound optimal suffers from high computational complexity, especially for large-scale scheduling problems. The algorithm's time and space complexity grow exponentially with the problem size, making it impractical for real-time or dynamic scheduling scenarios.

2. Random Scheduling:

Random Scheduling represents a simple and straightforward approach to task scheduling, where tasks are assigned to execution slots randomly without considering any optimization criteria. In this approach, tasks are selected for execution in a random order, leading to unpredictable scheduling outcomes.

While Random Scheduling may offer simplicity and ease of implementation, it typically results in suboptimal schedules with poor performance metrics. The lack of systematic optimization leads to inefficient resource utilization, increased makespan, and potential conflicts or bottlenecks in the execution sequence.

Random Scheduling is often used as a baseline or comparison method in task scheduling research to evaluate the performance of more sophisticated algorithms. However, its practical utility is limited in real-world scenarios where efficient resource allocation and scheduling optimization are critical.

3. Earliest Release Time (ERT):

Earliest Release Time (ERT) is a heuristic scheduling rule that prioritizes tasks based on their earliest available start time. In ERT scheduling, tasks are scheduled to start as soon as they become available for execution, without considering other optimization criteria such as task duration or resource constraints.

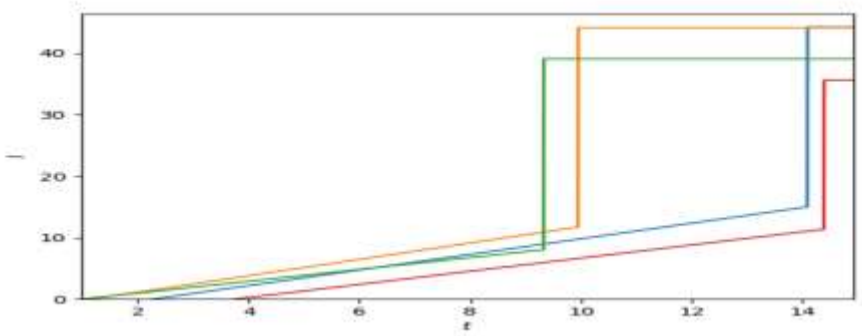
ERT scheduling aims to minimize waiting times and maximize resource utilization by starting tasks as early as possible. By scheduling tasks according to their release times, ERT helps ensure timely completion of tasks and reduces overall latency in the scheduling process.

VI. TASK SCHEDULING - REINFORCEMENT LEARNING FOR REAL-TIME SCHEDULING

Task objects expose following attributes

- name: task name
- t_release: earliest time at which a task may be executed
- duration : time required to execute the task
- is_important : bool
- has_intrest : bool

The tasks objects implement __call__ method that provides a monotonic non decreasing loss function quantifying the penalty for delayed execution.



Tasks – time vs loss for delayed execution

Quantifying the loss for delayed execution

Loss for the delayed tasks can be quantified using following attributes of the tasks determined by the priority.

Slope : defines the rate at which loss increases before the drop timet_drop, it determines the steepness of the linear increase in loss function.

t_drop : marks the time at which drop occurs in loss function. Beyond this point, the loss incurred by delaying task execution sharply increases.

l_drop : represents the magnitude of the drop in the loss function after t_drop. It specifies the loss incurred for delayed execution beyond the drop time.

VII. ACTOR-CRITIC ALGORITHM FOR TASK SCHEDULING OPTIMIZATION

The Actor-Critic algorithm is a reinforcement learning technique used to optimize task scheduling in dynamic environments. This algorithm consists of two main components: the actor, which learns to select actions (i.e., task scheduling decisions), and the critic, which evaluates the chosen actions and provides feedback to the actor.

The following content outlines the Actor-Critic algorithm used for task scheduling optimization:

1. Algorithm Overview: The Actor-Critic algorithm aims to learn an optimal policy for task scheduling by iteratively updating the actor and critic networks based on feedback from the environment. The actor network selects actions (i.e., task scheduling decisions) based on the current state of the environment, while the critic network evaluates the chosen actions and provides feedback on their quality.
2. Network Architecture: The Actor-Critic algorithm utilizes a neural network architecture comprising several components:
    a. Features Extractor: Extracts relevant features from the state of the environment, including information about available tasks and resource availability.

b. MLP Extractor: Processes the extracted features using multi-layer perceptron (MLP) layers, including a shared network for feature processing and separate networks for policy and value estimation.

c. Action Network: Outputs probabilities for selecting different actions (i.e., task scheduling decisions) based on the processed features.

d. Value Network: Estimates the expected value of selected actions, providing feedback to the actor network.

3. Training Procedure: During training, the Actor-Critic algorithm interacts with the environment by selecting actions based on the current state and receiving feedback on the chosen actions' quality. The actor network updates its parameters to improve action selection probabilities based on the received feedback (policy gradient update). The critic network updates its parameters to improve value estimation accuracy based on the observed rewards (value function update). Training proceeds through multiple iterations, with the actor and critic networks continually adjusting their parameters to maximize task scheduling performance.

4. Action Selection and Value Estimation: The actor network selects actions probabilistically based on the output of the action network, which represents the probability distribution over possible actions. The critic network estimates the expected value of selected actions, representing their potential impact on task scheduling performance. This value estimation guides the actor's decision-making process.

5. Optimization and Convergence: The Actor-Critic algorithm employs optimization techniques such as stochastic gradient descent (SGD) to update the actor and critic network parameters. Convergence of the algorithm is achieved when the actor learns an optimal policy for task scheduling that maximizes performance metrics such as makespan reduction and resource utilization efficiency.

6. Integration with Task Scheduling Environment: The Actor-Critic algorithm interacts with a task scheduling environment, which provides information about available tasks, resource constraints, and current scheduling decisions. The algorithm learns to adapt its scheduling decisions based on environmental dynamics, task characteristics, and performance feedback.

VIII. RESULTS

Conventional scheduling methods often face either excessive computational burdens or subpar performance. Emerging algorithms, which derive insights from similar problems, demonstrate the potential to generalize effectively, yielding nearly optimal schedules within a more manageable timeframe, thus enhancing practicality.

The initial scatter plot illustrates pairs of loss and runtime data in their raw form, each corresponding to a scheduling problem. Meanwhile, the subsequent plot displays excess loss values compared to the optimal solution. The Markdown table offers the mean values for these metrics.

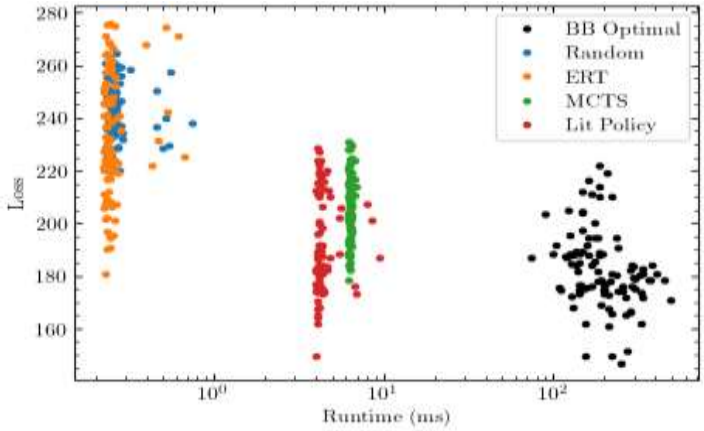


Fig 1. Loss Vs Runtime

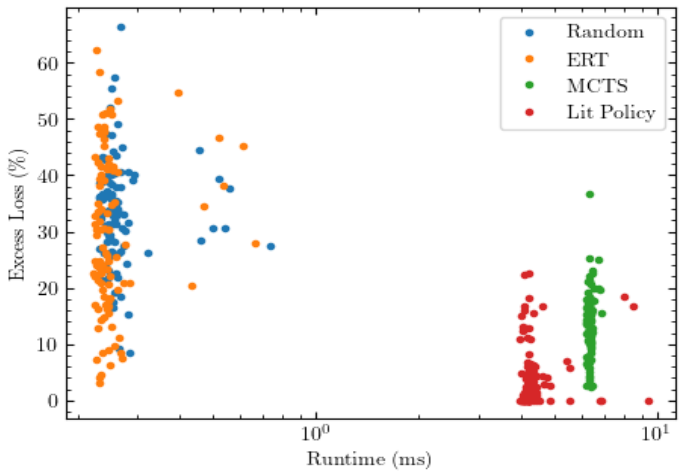


Fig 2. Excess Loss Vs Runtime

	Excess Loss (%)	Loss	Runtime (ms)
BB Optimal	0.000	182.440	207.535
Random	33.041	242.720	0.274
ERT	28.525	234.481	0.260
MCTS	13.284	206.675	6.340
Lit Policy	4.188	190.079	4.438

IX. CONCLUSION

In conclusion, our study comprehensively evaluated traditional task scheduling algorithms, including Optimal, Branch and Bound (B&B), Brute force, Random Sequencer, and Earliest Release Time (ERT), in a simulated environment. Through extensive experiments, we analyzed the performance of these algorithms in terms of make pan, resource utilization, and scheduling efficiency across various task distributions and system configurations.

Our findings highlight the trade-offs between computational complexity and scheduling performance in traditional algorithms. While Optimal and B&B algorithms offer near-optimal solutions, they incur high computational overhead, limiting their scalability in dynamic environments. The Brute force algorithm guarantees optimal solutions but becomes impractical for large-scale scheduling problems due to its exponential time complexity. Random Sequencer performs sub optimally compared to systematic optimization algorithms. In contrast, the ERT algorithm achieves competitive performance with reduced computational overhead, making it suitable for real-time scheduling applications.

These results underscore the importance of considering both performance metrics and computational efficiency when selecting task scheduling algorithms. While optimal solutions may be desirable, practical considerations such as computational resources and real-time constraints must also be considered.

Moving forward, future research could explore hybrid approaches that combine traditional and learning-based techniques to achieve optimal scheduling performance with reduced computational overhead. Additionally, investigating parallelization and distributed computing techniques could enhance the scalability of traditional algorithms for large-scale scheduling problems.

Overall, our study provides valuable insights into the strengths and limitations of traditional task scheduling algorithms, paving the way for the development of more efficient and scalable scheduling methodologies in dynamic environments.

References

[1]“Automated Time Manager: Effectiveness of Self-Regulation on Time Management through a Smartphone Application” 10.1109/ACCESS.2019.2926743, IEEE Access.

[2] “ScheduleME - Smart Digital Personal Assistant for Automatic Priority Based Task Scheduling and Time Management” 2021 2nd Global Conference for Advancement in Technology (GCAT) Bangalore, India. Oct 1-3, 2021.

[3] R. V. Adams and E. Blair, "Impact of Time Management Behaviors on Undergraduate Engineering Students’ Performance - Richelle V. Adams, Erik Blair, 2019", SAGE Journals, 2019.

[4] “Developing Real-Time Scheduling Policy by Deep Reinforcement Learning” 2021 IEEE 27th Real-Time and Embedded Technology and Applications.

[5] Nominal Batbaatar, Grace Amin (2021), “Students’ Time Management During Online Class”, Research Gate, pp 189.

[6] Mind Tools Content Team, “What Is Time Management? Working Smarter to Enhance Productivity”, accessed on November 22, 2021 at 02.47.

[7] J. Anderson and A. Srinivasan, “Early-release fair scheduling,” in Proceedings 12th Euromicro Conference on Real-Time Systems. Euromicro RTS.

[8] Ravichandran S. (2018) “Hands-on Reinforcement Learning with Python: Master Reinforcement and Deep Reinforcement Learning Using OpenAI Gym and TensorFlow”, Packt Publishing Ltd.

[9] Scott Fujimoto, Herke van Hoof, David Meger (2018) “Addressing Function Approximation Error in Actor-Critic Methods” Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

[10] Zhiping Peng, Delong Cui, Jinglong Zuo, Qirui Li, Bo Xu, Weiwei Lin (2015) “Random task scheduling scheme based on reinforcement learning in cloud computing” Cluster Comput (2015) 18:1595–1607DOI 10.1007/s10586-015-0484-2.