TODO Application

Ashish Bhat
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia, USA
ashishbhat@vt.edu

Disha Bhan
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia, USA
dishab2124@vt.edu

Humaid Desai
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia, USA
humaiddesai@vt.edu

Sahana Basapathi
Department of Computer Science
Virginia Polytechnic Institute and State University
Blacksburg, Virginia, USA
sahanabs@vt.edu

1 ABSTRACT

Time management is a virtue possessed by select few. Software engineers especially need to learn this because of the amount of work they are assigned to on a normal day in the office. A lot of engineers struggle with managing tasks in hand. They get overwhelmed and sometimes take hours to prioritize and properly complete their work. A plethora of tasks make engineers end up forgetting or delaying them because of their sheer amount. In addition to this, people with attention deficit hyperactivity disorder also find it difficult to focus on one task at a time. Todo app will help people with this condition as well as software engineers in organizing their thoughts by prioritizing their work. It will also help categorize the type of work which will give the user a coherent idea of the tasks remaining. The Todo app will help the user in decluttering their thoughts and providing them with a sense of achievement, helping them improve their productivity.

2 INTRODUCTION

Since the pandemic, work from home has given software engineers the opportunity to work in an unsupervised environment at their own pace. However, this convenience also caused many inconveniences to engineers with the most prominent being task management. Engineers have many meetings and upcoming deadlines for their work, and since they are not in an office environment it gets challenging to keep track of all the things. Hence there is an inherent need for a Todo app that will help software engineers organize their tasks in a systematic way so that it can maximize their productivity and prevent them from missing any of the tasks or deadlines. This problem is worth solving because efficiency is an important part of any software engineer's work and if they are not motivated and satisfied by their work, it can negatively impact their productivity which may in turn decrease their work performance. Employees, especially software engineers should be constantly motivated because they have to deal with a lot of rigorous and redundant tasks such as finding bugs, meetings, and feedbacks from the client. Todo lists are important for anyone looking to accomplish various tasks within a set period. In a nutshell a Todo app offers systematic management of tasks which helps in improving the productivity of people especially those who have ADHD, it helps in providing a sense of completion which in turn motivates people to complete their other tasks as well, and finally it helps in meeting deadlines, prioritizing and categorizing the work.

3 RELATED WORK

Lunatask.app is an all-in-one privacy focused Todo list which sorts the tasks that are input, based on age, priority and estimated time needed to complete the task. This was the base upon which we augmented the idea of grouping the tasks according to categories.

A publication by Victoria Bellotti et al. [1] sought to discover what kinds of task management demands might be supported by a task list manager system. They proposed that the principal problem of task management was not poor prioritization, but the effort it required. They outlined resources and methods people used that helped ensure that they were effective at this. Based on this feedback and research, they designed TaskVista as a tool to reduce that effort. We have incorporated the feedback given by users in this paper as a guide while making our application.

4 DESIGNS

4.1 ARCHITECTURAL COMPONENTS AND THEIR INTERACTIONS

Model-View-Controller architecture is used to structure this system. This application will have a UI to interact with the users along with storage and retrieval of data. In MVC architecture there are three components: model, view, and controller and these are built to handle specific development aspects of an application. It helps with faster development as the programmers can work parallel to develop several components. For Instance, for a TODO app system, one programmer can work on the view while the other can work on the controller to create the business logic for the system. With this we can make scalable and extensible projects. The architecture we plan to use for this project is detailed in the above figure. The components are explained below: **Model**: Corresponds to all the data-related logic that the user works with. We plan to use the postgreSQL database since it is a free, open-source, highly customizable database. PostgreSQL also

has great support for JSON data. There is also JSONB data type added, which improves the indexing ability. It takes the query as the inputs and returns the task lists accordingly.
View: The view component will contain the frontend or the UI logic such as displaying a to-do list, having an editable view for each to-do task, and interface for creating a new task. Appropriate view according to the task request to the user. We also plan to have an activity feed to check other's activities.
Controller: The controller will be the main component that will bridge the gap between the model and the view component by processing input and formulating outputs. It controls the interaction between Model and View, where the view calls the controller to update the model. A view can call multiple controllers if needed. Here we define the business logic. Eg, filter the task list according to the user, check if the user has appropriate permissions to change and update the tasks.

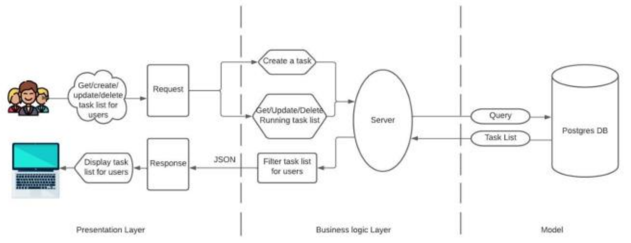

Get/create/ update/delete task list for users
Request
Create a task
Get/Update/Delete Running task list
Server
Query
Task List
Postgres DB
Display task list for users
Response
JSON
Filter task list for users
Presentation Layer
Business logic Layer
Model

Figure 1: High Level Architectural Design

4.1 CONSTRAINTS OR GUIDELINES

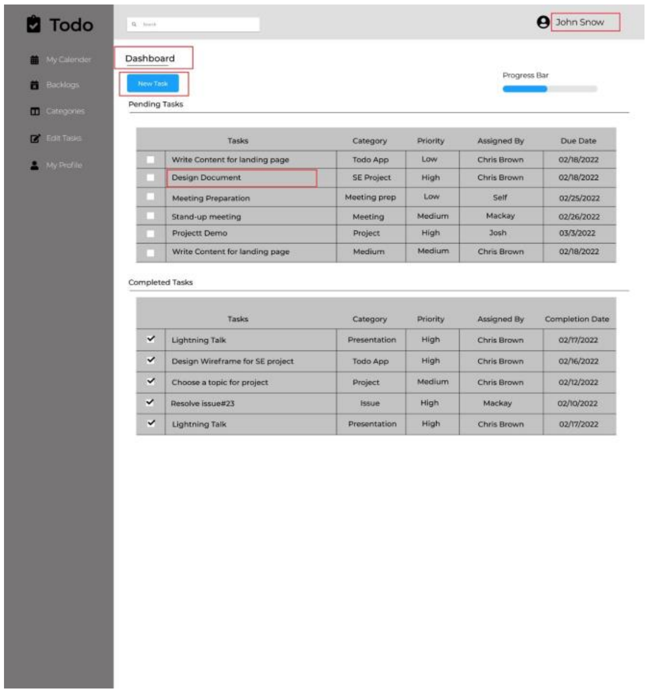
Some of the constraints we foresee in building the software are in the usage of Google Calender api. This Calendar API has a limit of 1,000,000 queries per day. This could be a constraint if that feature of the web application is used a lot. It also creates a point of dependency on the working of the api. A failure of the api can make the feature display an error message. In addition to the api constraints, we plan to use django framework to build the web application. There are some aspects of Django that are hard to optimize. For example, we will always have to wait for Django when it is running requests through middleware, serializing and deserializing JSON strings, converting database queries into Python objects and running garbage collection, etc. This can make application a little slower.

4.3 ADDITIONAL DESIGN PATTERNS

Creational design patterns might be useful for implementing this project. We can create only one instance of a class and provide only one global access point to that object. This can ensure there is one point of access to the to-do list class, and reusability of code. Like google calendar, a todo list should be the same independent of the devices though which it is accessed (global access to the object), thus ensuring consistency of data.
In the creational design pattern, we plan to use the singleton pattern is a software design pattern that restricts the instantiation of a class to one "single" instance. This is useful when exactly one object is needed to coordinate actions across the system. Builder pattern may also be used here. This is because since a to-do application would require the creation of many tasks, these tasks will have the same code i.e., creating a task for meeting preparation will not be different from creating a task for studying. This will increase flexibility as a lot of the code will be the same and only the semantics of the task will change. The APIs, task creation, editing, and deleting will be the same for all the to-do tasks in the list.

4.3 DESIGN SKETCHES

The prototype of the project can be accessed using this link. With the screenshots below, we detail how to navigate and understand the flow of the application.
Fig 2 shows the snapshot of the dashboard of the application. It consists of the task, along with the category, respective priority, who it is assigned to and the due dates. There is also a list of completed tasks. This view can be seen by clicking on the "Dashboard".



Pending Tasks

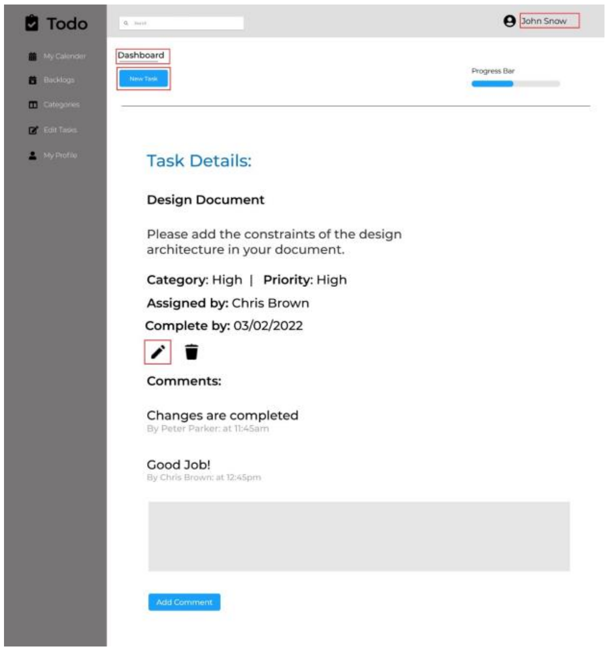
	Tasks	Category	Priority	Assigned By	Due Date
☐	Write Content for landing page	Todo App	Low	Chris Brown	02/18/2022
☐	Design Document	SE Project	High	Chris Brown	02/18/2022
☐	Meeting Preparation	Meeting prep	Low	Self	02/25/2022
☐	Stand-up meeting	Meeting	Medium	Mackay	02/26/2022
☐	Projectt Demo	Project	High	Josh	03/3/2022
☐	Write Content for landing page	Medium	Medium	Chris Brown	02/18/2022

Completed Tasks

	Tasks	Category	Priority	Assigned By	Completion Date
✓	Lightning Talk	Presentation	High	Chris Brown	02/17/2022
✓	Design Wireframe for SE project	Todo App	High	Chris Brown	02/16/2022
✓	Choose a topic for project	Project	Medium	Chris Brown	02/12/2022
✓	Resolve issue#23	Issue	High	Mackay	02/10/2022
✓	Lightning Talk	Presentation	High	Chris Brown	02/17/2022

Figure 2: Overview of Dashboard

Fig 3 lists the task details. This can be assessed by clicking on the respective task from the dashboard.



Task Details:

Design Document

Please add the constraints of the design architecture in your document.

Category: High | **Priority:** High

Assigned by: Chris Brown

Complete by: 03/02/2022

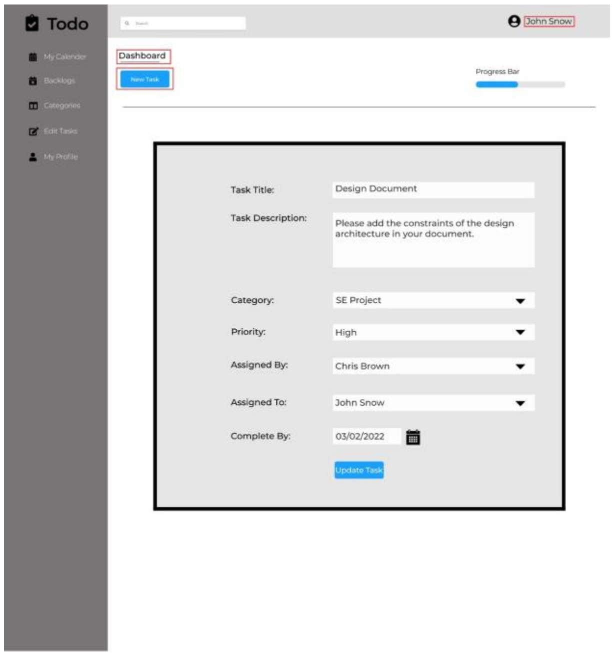
Comments:

Changes are completed
By Peter Parker: at 11:45am

Good Job!
By Chris Brown: at 12:45pm

Figure 3: Details of a task

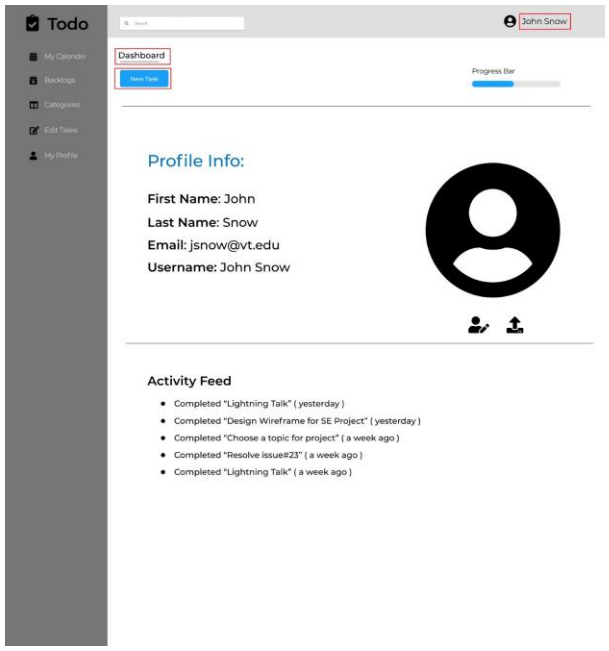
Fig 4 is the screenshot that is seen by a user to edit a task. Any task can be edited by using the edit button from the task detail's view.



Task Title:	Design Document
Task Description:	Please add the constraints of the design architecture in your document.
Category:	SE Project
Priority:	High
Assigned By:	Chris Brown
Assigned To:	John Snow
Complete By:	03/02/2022

Figure 4: Updating the task

Fig 5 shows the profile information and the activity feed of the user. Activity feed shows tasks shared with the user, user's concurrent tasks to be completed, etc.



Profile Info:

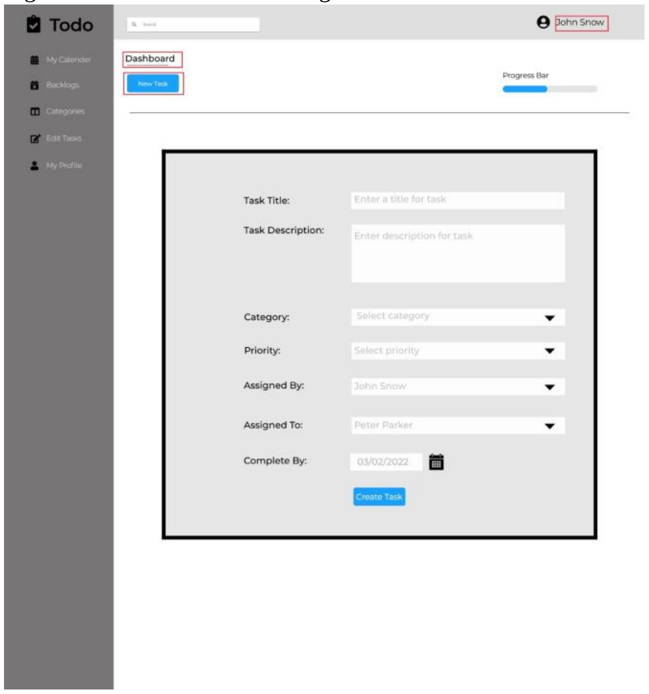
First Name: John
Last Name: Snow
Email: jsnow@vt.edu
Username: John Snow

Activity Feed

- Completed "Lightning Talk" (yesterday)
- Completed "Design Wireframe for SE Project" (yesterday)
- Completed "Choose a topic for project" (a week ago)
- Completed "Resolve issue#23" (a week ago)
- Completed "Lightning Talk" (a week ago)

Figure 5: Overview of the profile of a user

Fig 6 shows the form that a user gets to create a new task.



Task Title:	Enter a title for task
Task Description:	Enter description for task
Category:	Select category
Priority:	Select priority
Assigned By:	John Snow
Assigned To:	Peter Parker
Complete By:	03/02/2022

Figure 6: Creating a task

REFERENCES

[1] Bellotti, Victoria & Dalal, Brinda & Good, Nathaniel & Flynn, Peter & Bobrow, Daniel & Ducheneaut, Nicolas. (2004). What a to-do: studies of task management towards the design of a personal task list manager. Proceedings of CHI, Vienna, Austria. 24–29. 735-742. 10.1145/985692.985785.