Design Document

TaskX - “Get organized. Work smarter. Stay Motivated”

Table of Contents

1. Introduction

1.1 Purpose of this document

- The purpose of this Software Design Document is to provide a description of the design of a system fully enough to allow for software development to proceed with an understanding of what is to be built and how it is expected to be built.

1.2 Identification

- The software will be using localhost to access the network services that are running on the host via the loopback network interface. MongoDB is used for database design.

1.3 Scope of the Document

- This design document provides a description and overview of the technical design for TaskX. This document gives a compositional outline of the framework to portray various parts of the system. This document likewise works as a primary reference point for designers. It presents various diverse engineering perspectives to portray various parts of the system. It is proposed to catch and pass on the critical compositional choices which have been made on the system. The essential target group of this report are system designers and system developers.

1.4 Targeted Audience

- This project is targeted to be used by people who lack motivation to study continuously with full focus. This SRS is intended for several audiences, including the customer, as well as the project manager, designers, developer and tester.
- The client will utilize this SRS to check that the designer group has made a system that is adequate to the client.
- The project manager of the developer team will use this SRS to plan milestones and delivery date and ensure that the developing team is on track during the development of the system.
- The designers will utilize this SRS as a reason for making the System's design plan. The designer will constantly refer back to this SRS to guarantee that the framework they are planning will satisfy the client's necessities.
- The developer will utilize this SRS as a reason for building up the system's usefulness. The developer will interface the prerequisites characterized in this SRS to the product

they make to guarantee that they have made programming that will satisfy the entirety of the client's archived necessities.

- The tester will utilize this SRS to determine test plans and experiments for each document requirement. At the point when parts of the product are finished, the tester will run his tests on that product to guarantee that the product satisfies the requirement document in this SRS. The tester will again run his tests on the whole system when it is finished and guarantee that all requirements reported in this SRS have been satisfied.

1.5 Key Stakeholders

- A good discovery process is critical to software development. The requirements generated here set the stage for the entire project, laying the groundwork for success or failure. The term “stakeholder’ refers to the people or groups affected by a software development project. Stakeholders exist both within the organization and outside of it. Here is the list of stakeholders connected to the project:
    - Students/ Professionals
        - The system helps the students and professionals in creating schedules for certain tasks which include all sorts of activities like Homework, Assignments, Exams, Deadlines, Meetings and so on depending on the profession of the individual.
    - Developers
        - They build the software based on feedback from other stakeholders, but they’re also stakeholders in their own right. They have the technological expertise necessary to advise executives on which features are feasible and how long each would take to build.
    - Content Writers
        - Researching industry-related topics (combining online sources, interviews and studies) Writing clear marketing copy to promote our products/services. Preparing well-structured drafts using Content Management Systems.

1.6 Languages and Tools

- Node JS
- React JS
- Express JS
- VS Code
- Postman
- MongoDB
- VisualParadigm
- Draw.io
- Canva
- CSS

2. Design Considerations

2.1 Goals and Guidelines

2.1.1 Architecture:-

- The system must be able to satisfy all the functional needs as desired by the user along with having the non functional requirements taken into consideration while developing the solution. The system should be made in such a way that additional functionalities or use cases should be easily incorporated in the future.

2.1.2 Development Environment:-

- The development of the system or the application must be consistent. With the help of technological advancement and its efficient usage, aim is to build a system which should support the feature of backward compatibility in order to achieve better performance.

2.1.3 Ease of Use:-

- The functions of the system or the application must be user friendly and the user must not have any hard time using the system. The system aims at helping the users of the system with managing their daily tasks in an effective manner and the user would want this function to work with ease and provide output in a minimum amount of time.

2.2 Operational Environment

- MongoDb for Database
- Github Repository
- Node JS
- Express JS
- React JS
- UI/UX Interface

Functional Goals of the system :

- Web application with the latest technologies used
- High performance of the system
- Efficient sharing of data with the help of appropriate data structures.

2.3 Development Methods

- **Security :-** The system architecture should be designed in such a way that only a minimal amount of information or the code is visible to the user. A proper abstraction to the back-end pieces should be provided and there should be various levels of the security for each of the subsystems or the functions.
- **Separation of Responsibility :-** The system should be designed in a modular fashion such that each piece of the code has a set of responsibilities associated with it.
- **Restful Framework :-** REST API is a plain and simple framework which provides the user with the feature of working with flexibility. The framework is not dependent on any specific programming language. The system architecture should be able to replace each of the layers independently and even use different languages that might be better suited for a certain layer
- **Functionality :-** The software is capable of providing the user with the functions which meets the stated needs when the software is used under some specified conditions
- **Reliability and Usability :-** The software is capable of maintaining the level of its performance under stated conditions for some period of time. Also it is capable of Understanding , learning, usage and seems attractive to the users.
- **Maintainability :-** Capability of the software being modified. The modifications must include the corrections, improvement of the software to all the changes in the environment

2.4 Development Environment

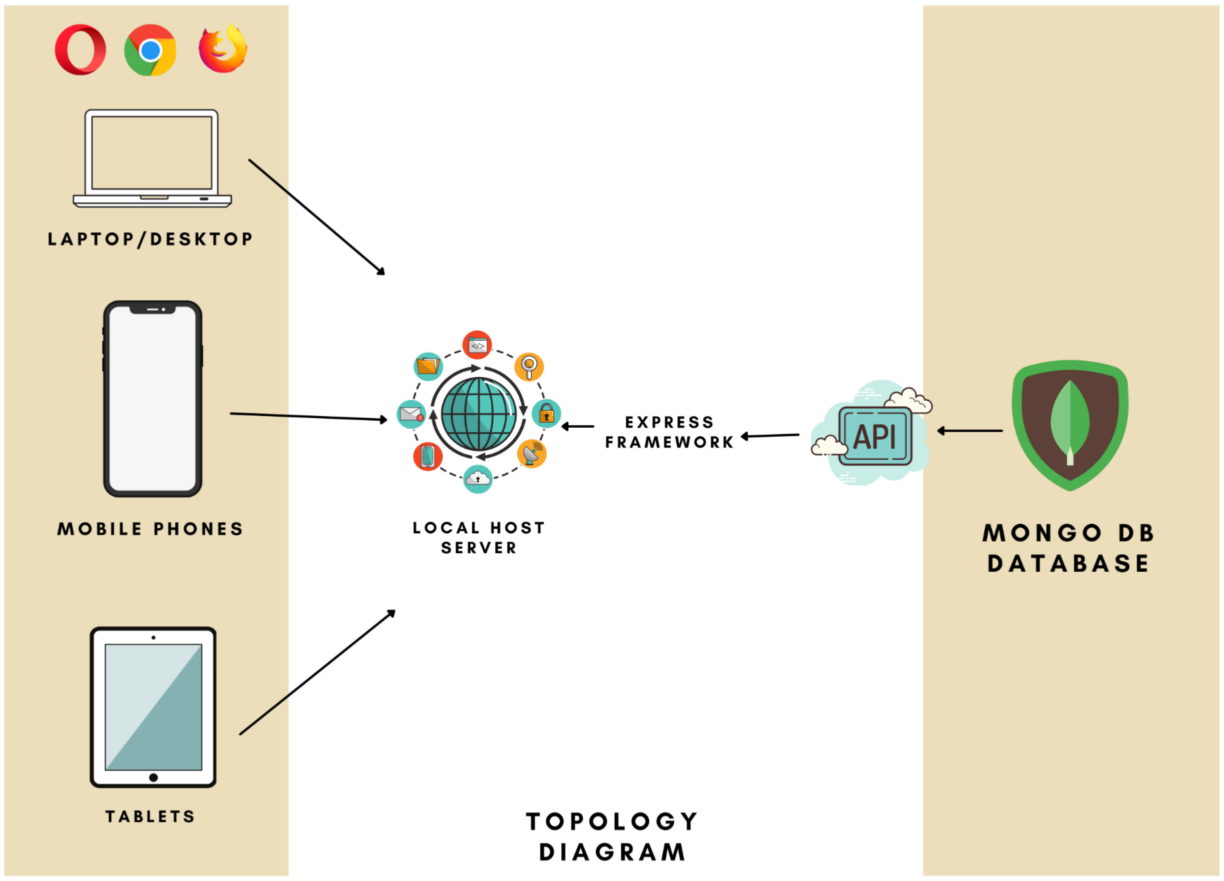
Software	**Description**
MongoDB	Database
Github	Version Control repository
Node JS	Programming language for the server side
React JS	Programming language for client side
Express JS	Backend Framework

3. System Architecture and Architecture Design

- The proposed section outlines the system architectural design.

3.1 Topology Diagram:

- Network topology is the way a network is arranged, including the physical or logical description of how links and nodes are connected to each other and how communication takes place among the networks.

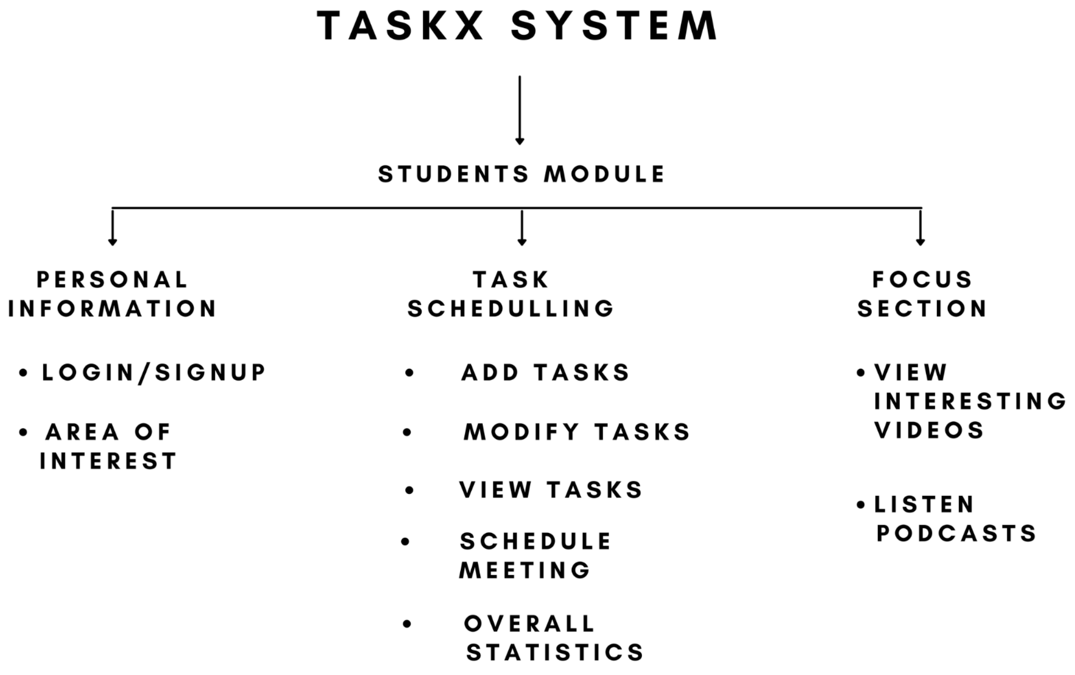


3.2 System Architecture

- A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system.

3.2.1 Website Architecture Diagram

- A hierarchical structure is used to visualize the website's overall framework. A user can see the directory structure of the web pages and organization of the website content.

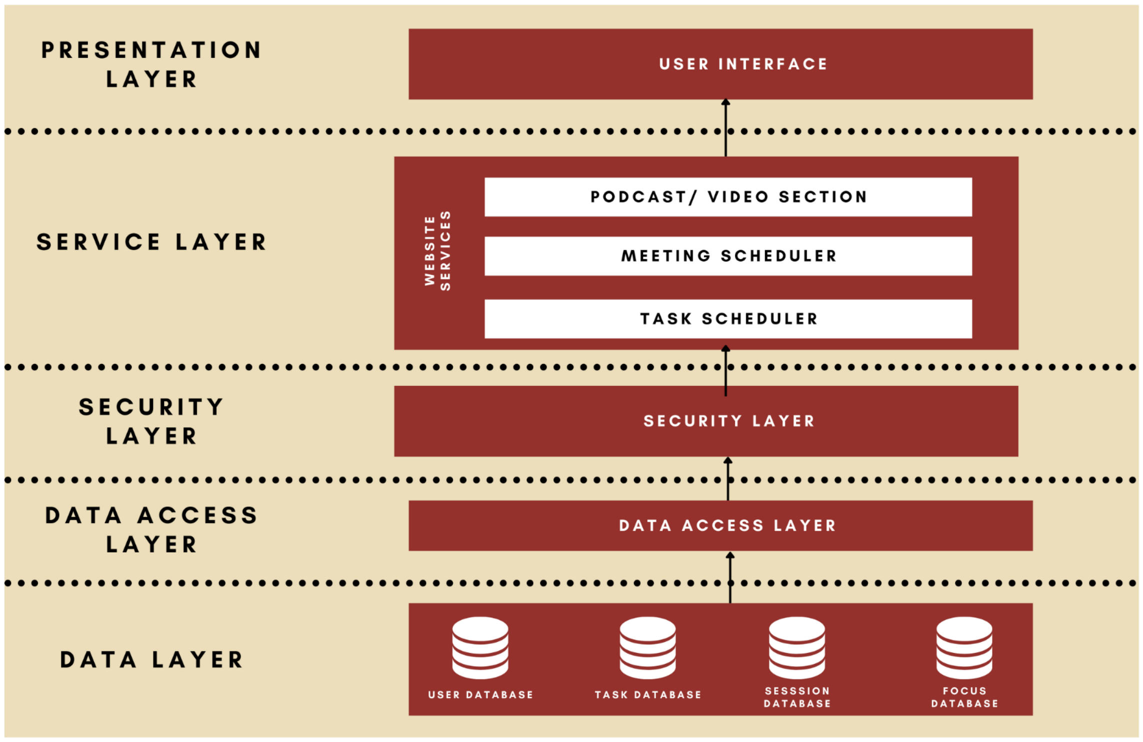


WEBSITE ARCHITECTURE DIAGRAM

3.3 Software Architecture

3.3.1 Software Architecture Diagram

- An **architectural diagram** is a **diagram** of a system, used to abstractly visualize the overall outline of the **software** system and the relationships, constraints, and boundaries between components. One can use architecture diagrams to describe patterns that are used throughout the design process.



SOFTWARE LAYERED ARCHITECTURE DIAGRAM

3.3.2 Software Element:

	System	Description
Programming Language	JavaScript	**JavaScript** is a prototype-based, multi-paradigm, single-threaded, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming)

		styles.
Frontend Framework	Node/React.JS	**React.js** is a JavaScript library for building modular i.e. component based user interfaces. React can render on the server using Node.
Backend Framework	Express.JS	**Express.js** is a fast, unopinionated, minimalist backend web framework for Node.js. Express.js supports development on both, server and user side.
Database	MongoDB	**MongoDB** is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

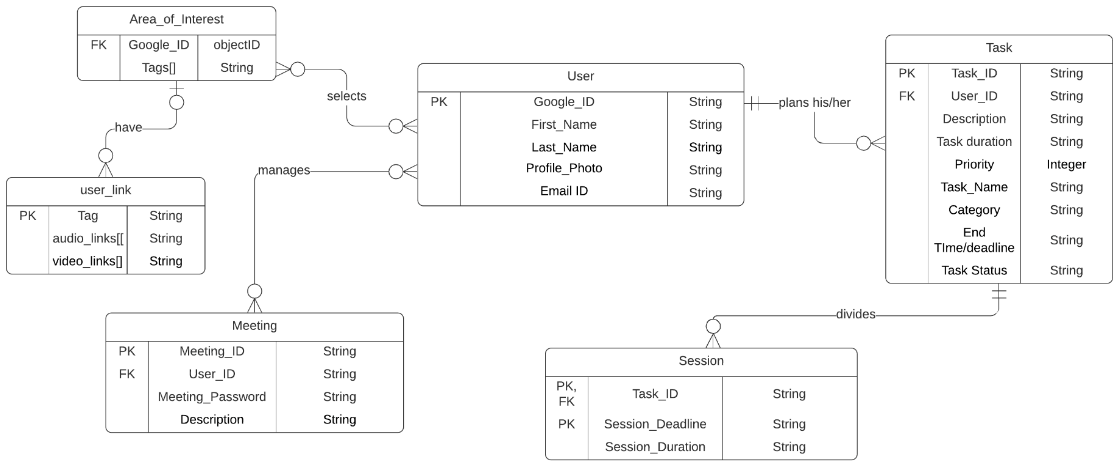
4. System Design

The proposed system aims at providing a framework to plan a user's task i.e. a user can add his daily fixed tasks, daily dynamic tasks, meetings, etc using the system. Users can also view motivational videos and hear podcasts listed according to his interests. Proposed design may be developed to incorporate modular components that plugs into the application. The system will try to provide strong security to ensure privacy of the users.

4.1 Business Requirements

- User Interface by which the user and a computer system interact, in particular the use of input devices and software
- User can login/signup into the platform using Gmail-ID and password
- Interaction modules by which use can interact with the system
- View only modules by which use can only view modules as a feedback from the system

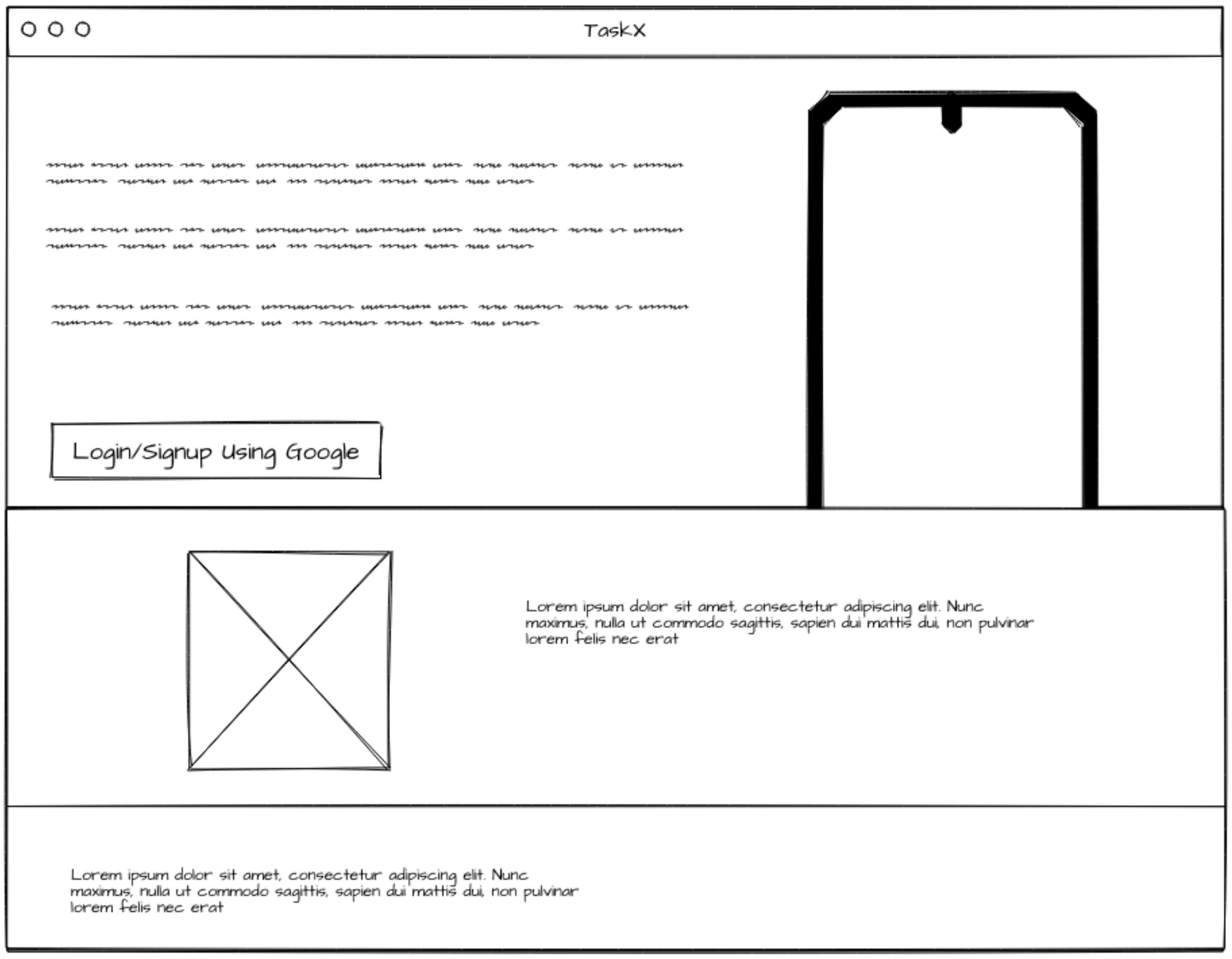
4.2 Database Design



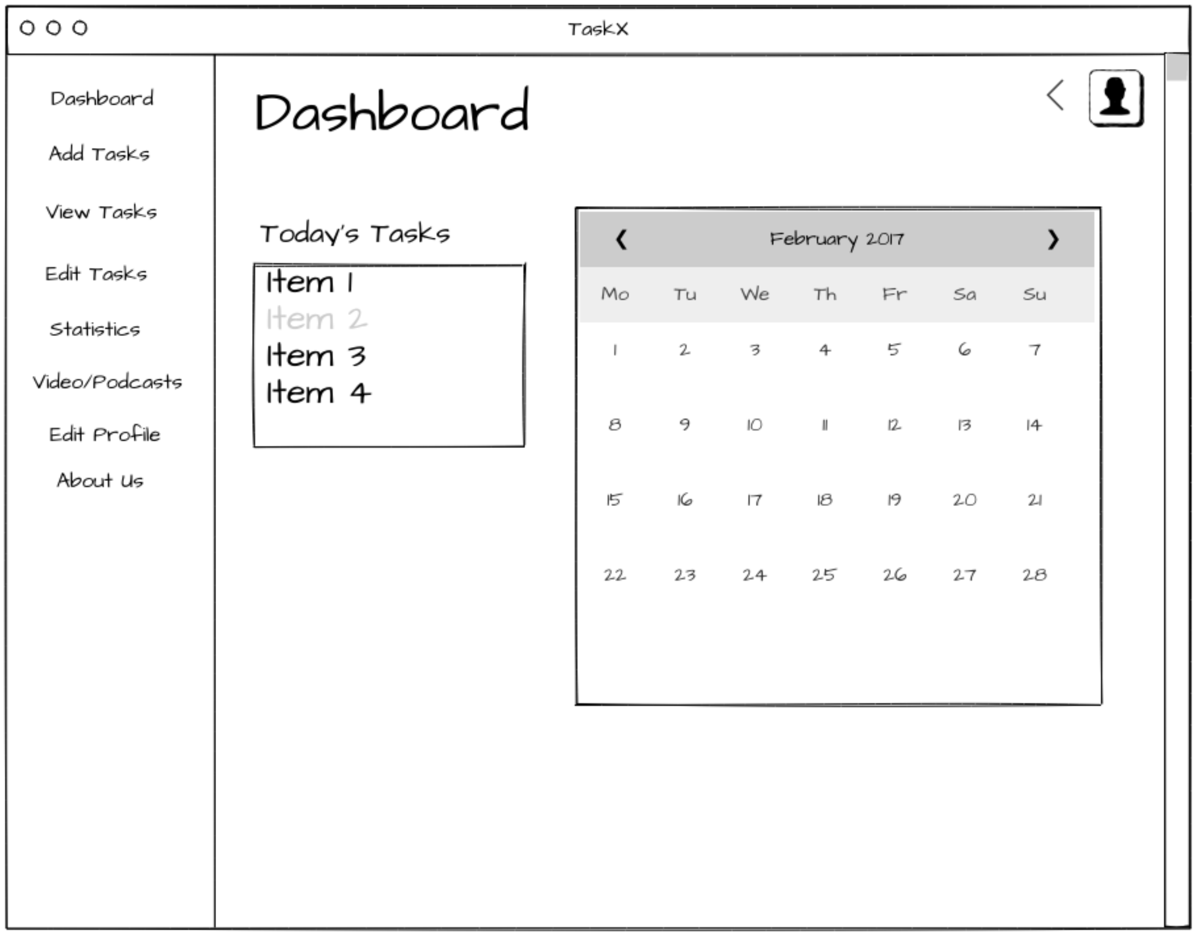
For a better view:
https://drive.google.com/file/d/1bXnXUF8mfcTXlPQ8Gt7fkjRDtAH_zpeo/view?usp=sharing

4.3 User Interface Design

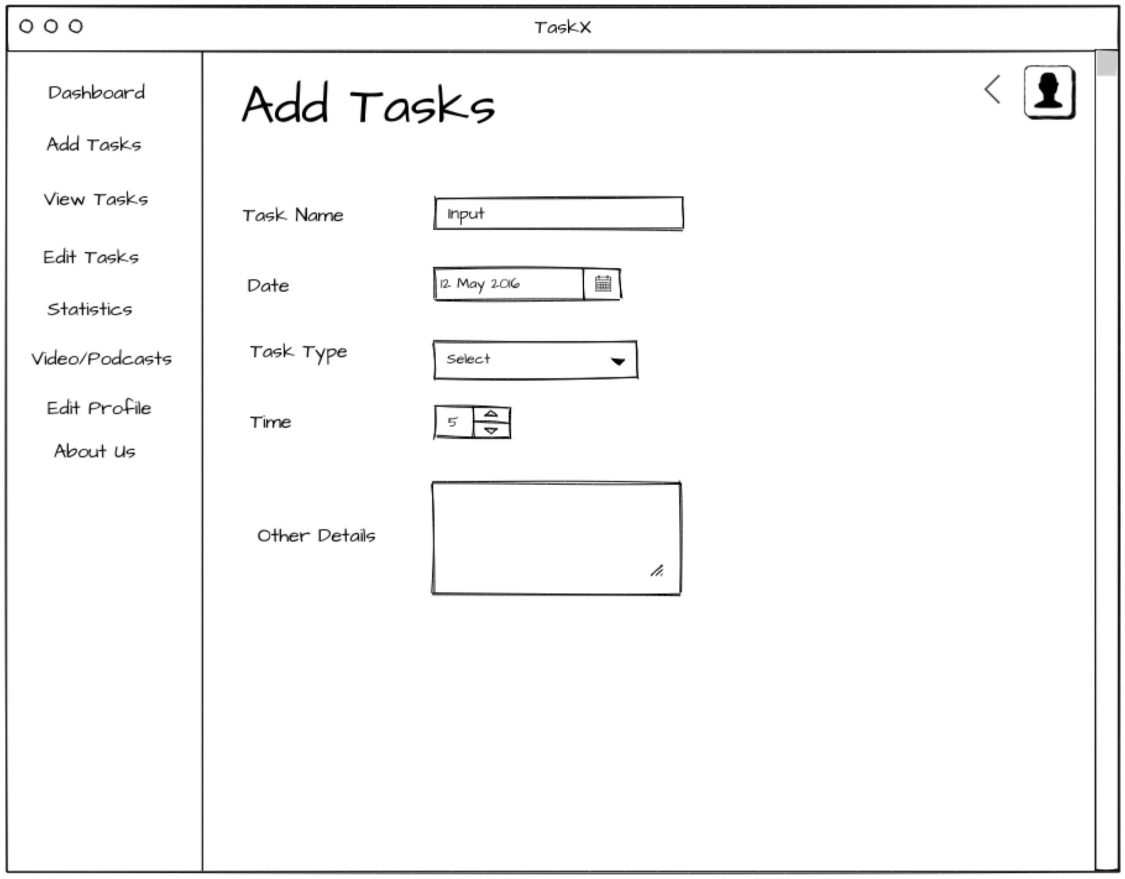
4.3.1 Login/SignUp page


TaskX
Login/Signup Using Google
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat

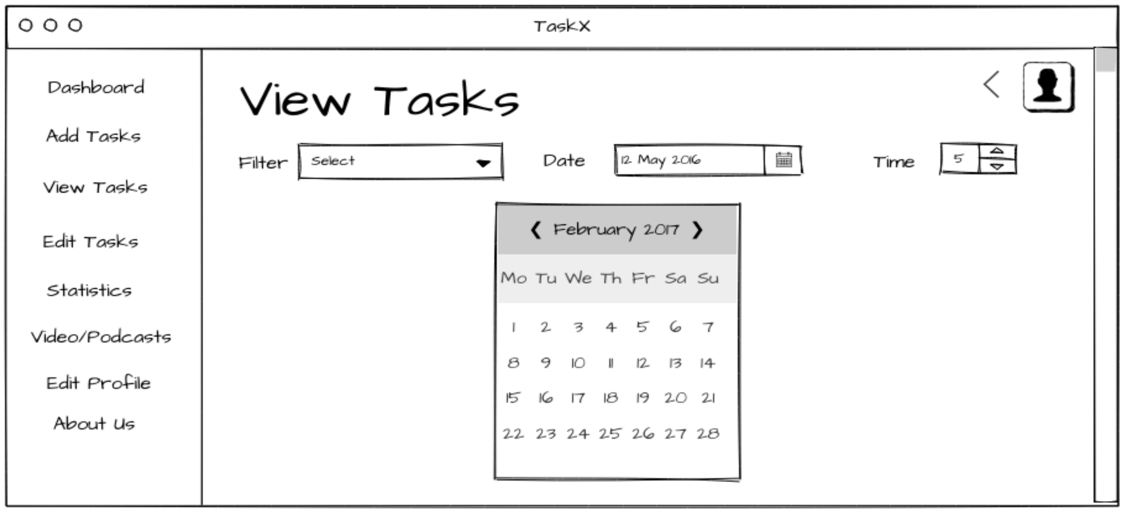
4.3.2 Dashboard



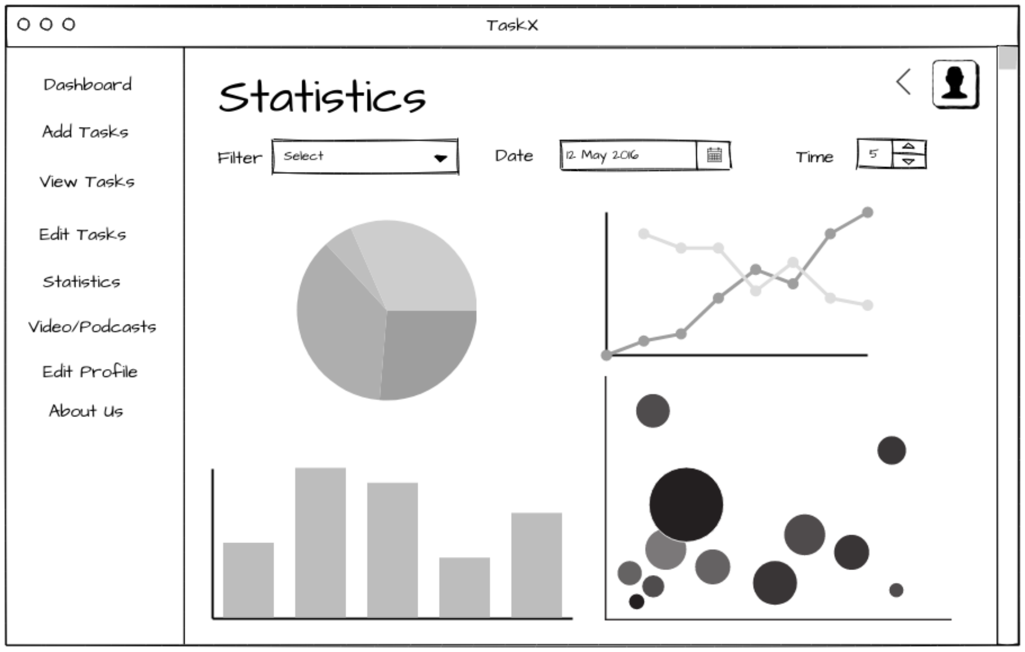
4.3.3 Add Task


TaskX
Dashboard
Add Tasks
View Tasks
Edit Tasks
Statistics
Video/Podcasts
Edit Profile
About Us
Add Tasks
Task Name
Input
Date
12 May 2016
Task Type
Select
Time
5
Other Details

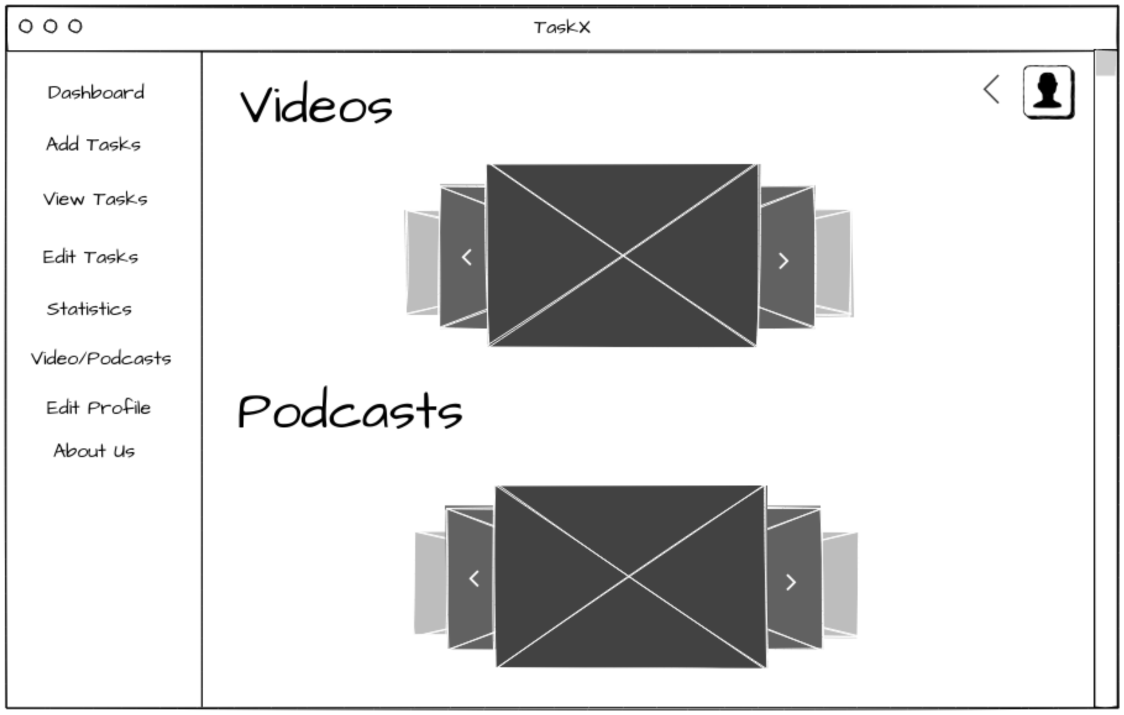
4.3.4 View Task



4.3.5 Statistics



4.3.6 Focus Section


TaskX
Dashboard
Add Tasks
View Tasks
Edit Tasks
Statistics
Video/Podcasts
Edit Profile
About Us
Videos
Podcasts