|  |  |
| --- | --- |
|  | untitled |

**Tasky**

Bachelor’s Thesis

Oğuz Kaan Yazan Berke Biten

217CS2019 217CS2029

Supervised by

Emine Ekin

June 2021

Abstract

Most of the people work with teams on their job or school. Following all the tasks and keep staying updated about the tasks is getting harder when the number of people that you work with and number of tasks that you worked on increase. Also, for the people who manage projects, getting reports about the project and the people who works on the project is really hard thing when you try to do it without using any software.

We designed a project and task management system that works on web browsers and smart phones to make working on a projects and tasks with the teams easier. The system is built top of two main concepts: Project and Task. Users can create projects and, in that project, they can create tasks. Each task has an assignee. Users can save their works on the tasks, list the tasks that is assigned to them. With that way planning the works gets easier. Also, there is a project report system which makes report about the project and team members for the project owners.

Acknowledgements

Thanks to designers of MS Office 365, it took a long time to prepare this template. I hope you will be able to use without destroying the format and organize the content.

If you would like to say thank you or to acknowledge some people, you can write into this part. Otherwise, remove this part completely.

Good luck.

Table of Contents

[1. INTRODUCTION 2](#_Toc61190346)

[1.1. Problem Definition //Style is Heading 2 2](#_Toc61190347)

[1.1.1. Heading 3 Example 2](#_Toc61190348)

[1.1.2. Heading 3 Another Example **Error! Bookmark not defined.**](#_Toc61190349)

[1.2. Using the Style Gallery **Error! Bookmark not defined.**](#_Toc61190350)

[1.2.1. Captioning Images and Tables **Error! Bookmark not defined.**](#_Toc61190351)

[1.2.2. Referencing the Figures and Tables **Error! Bookmark not defined.**](#_Toc61190352)

[1.2.3. Creating List of Figures, and List of Tables **Error! Bookmark not defined.**](#_Toc61190353)

[1.2.4. Preparing the References Section, and Citing **Error! Bookmark not defined.**](#_Toc61190354)

[1.2.5. Updating Table of Contents, List Of Figures, and List of Tables **Error! Bookmark not defined.**](#_Toc61190355)

[1.3. Organization of the Thesis 2](#_Toc61190356)

[2. LITERATURE REVIEW 3](#_Toc61190357)

[2.1. Heading 2 3](#_Toc61190358)

[2.2. Petri Nets 3](#_Toc61190359)

[3. PROPOSED SYSTEM 4](#_Toc61190360)

[3.1. Introduction 4](#_Toc61190361)

[3.2. Heading2 8](#_Toc61190362)

[3.3. Heading2 8](#_Toc61190363)

[3.3.1. Heading3 8](#_Toc61190364)

[3.3.2. Heading 3 8](#_Toc61190365)

[3.4. Heading 2 9](#_Toc61190366)

[4. Implementation, tests, experiments 10](#_Toc61190367)

[4.1. Implementation 10](#_Toc61190368)

[4.2. Tests 10](#_Toc61190369)

[4.3. Experiments 10](#_Toc61190370)

[5. conclusions and future work 11](#_Toc61190371)

[6. References 12](#_Toc61190372)

LIST OF FIGURES

[Figure 1.1 Displaying style gallery. **Error! Bookmark not defined.**](#_Toc61190373)

[Figure 1.2 Caption in style gallery. **Error! Bookmark not defined.**](#_Toc61190374)

[Figure 1.3 Insert caption option for images. **Error! Bookmark not defined.**](#_Toc61190375)

[Figure 1.4 Cross-Reference function in References tab. **Error! Bookmark not defined.**](#_Toc61190376)

[Figure 1.5 Adding new reference into list of references. **Error! Bookmark not defined.**](#_Toc61190377)

[Figure 1.6 Different types of sources. **Error! Bookmark not defined.**](#_Toc61190378)

[Figure 1.7 Citing the sources added earlier. **Error! Bookmark not defined.**](#_Toc61190379)

[Figure 1.8 Creating the "List of references". **Error! Bookmark not defined.**](#_Toc61190380)

[Figure 1.9 Updating the table of contents **Error! Bookmark not defined.**](#_Toc61190381)

[Figure 3.1 Error Values of Each Video for Number 8](#_Toc61190382)

LIST OF TABLES

[Table 2.1. Result on Bobot Dataset with Scale Grid Approach 3](#_Toc61190383)

# 

# INTRODUCTION

## Problem Definition

Most of the people work on multiple tasks every day in their job, school or even in home. Working on these tasks is getting harder if you try to handle it with a pen and paper. Also, for the people who are responsible for managing a team, it is hard to monitor the project, tasks and the team. Especially for the large teams it is almost impossible to follow the situation about the project and tasks.

## Purpose of the System

Purpose of the task management system Tasky is making easier and more efficient working and managing the projects and tasks. Users will easily create, assign and prioritize tasks, set deadlines, track how much time spent on tasks and visualize the projects and tasks with the Tasky.

## Scope of the Project

Task management system Tasky is designed for making working on a project easier and more efficient. The system consists of a web site and mobile application. Users can create projects in the web site and add participants to the project. Once the project is created users can create tasks within the project. Each task has a reporter and assignee. Assignee should be selected when the task is created. The reporter of the tasks is the user that is created the task. Users can also create sub-task which is related with any of the existing tasks.

There are three roles in a single project. Project manager, watcher and team member. Project manager is the user who is the created the project by default. Project managers can view the reports about the project, tasks and team members. Team members can log work to the tasks. Users whose have watcher role in the project can only monitor the project. They can not do any operation.

## Organization of the Thesis

The organization of this document is as follows:

Chapter 1 introduces the problem. For this template document only, table and image captioning, preparing the list of references, and citing a source, in general using the style gallery is also explained in Chapter 1. The relevant literature survey, existing similar tools and/or applications have been presented in Chapter 2. In the further sections, the design of the solution, testing, experimentations (particularly if you worked on a research problem), and conclusions have been presented.

# LITERATURE REVIEW

Section 2 of RAD, which is titled as Current System will be placed here. There is also a similar section in SDD, which should also appear in this chapter. **Note that, project supervisor may ask you to change chapter title, and contents.**

If your project is not a replacement of an existing system, a comparison between your system and similar systems’ properties might be included here.

If your project relies on some theoretical background, i.e., some well-known algorithms etc., it must be mentioned and explained in this section as well.

## Heading 2

Below, Table 2.1, you can find a table, and table caption.

Table 2.1. Result on Bobot Dataset with Scale Grid Approach

|  |  |  |
| --- | --- | --- |
| Seq | #Frame Size | Percentage Overlap |
| A.  B.  C.  D.  E.  F.  G.  H.  I.  Ja.  Jb.  K.  L. | 601  628  403  946  304  452  715  411  1016  387  387  1019  1307 | 61  83  92  80  85  64  74  96  83  83  80  84  85 |
| Avg. |  | 81 |

Table 2.1presents average percentages of overlap to the target objects on the Bobot dataset.

## Petri Nets

Petri Nets may seem to be a powerful alternative in modeling distributed systems, since they are specifically designed to model concurrent systems with interacting components. Petri Nets have been developed from the early work of Carl Adam Petri in his doctoral dissertation [3].

# PROPOSED SYSTEM

In this project, a project/task management system named Tasky is proposed. Tasky is designed to make easier and more efficient all the processes that can be done in a project by any member of the project team. Tasky system consists of a web application and mobile application.

For all the members of a project, there are different pages and screens that makes easier to follow the project and tasks. Users can easily monitor the project and tasks that is assigned to them with the task pages/screens. Users can display, filter, sort and search tasks and plan their daily works.

Also for the users who manages the project, there is a report page that summarize the project and providing the critical information about the project, task and project members.

## Introduction

Tasky system is built on top of three main entities: project, task and work log. Users can create projects. There are tasks under the projects and users can add tasks as many as they want. Project members log their work under tasks.

There are participants of the project. Each project participant has a role. There are three different roles within a project: Project manager, watcher and team member.

Project manager role is the owner of the project. There can be only single project manager in the project.

Team member role is the participants who works on the tasks.

Finally, there are watchers who can view the project pages but has no permission to do any operation.

### Project

Projects are created by registered users. All registered users can create projects. A project has title, description, status and files. There are also project participants who work on the project. Project participants can be invited when the project is created or after the project created.

### Task

In every project, there are tasks which are created by project manager or team member. A task has title, description, due date, status, assignee and reporter. Assignee and the reporter are one of the project participants. Assignee is who is the responsible of the project. Reporter is who will be notified about the task operations. Tracking of the tasks is done with the statuses of the tasks. There are 4 different statuses. To-do, active, resolved and closed. To-do status means the task is not started yet. Active status is means someone is working on the task. Resolved status means the task is finished but not closed/approved yet. Closed status means the task is completed.

### Work Log

Work log has duration, description and date. It represents the job that is done on the tasks. Work logs are created under the tasks by project manager or team member. Work logs are used when in the project reports that is visible to only project manager.

## Functional Requirements

### Registration

Users can register to the system by filling a registration form. They can login to the system with e-mail and password information. After they logged in to the system, they have a profile created by their registration credentials. They can view and edit their profile. Users also can manage their preferences which includes turn on/off notifications and change password functions. These functions are available in both platforms.

### Project Management

Users can create and manage projects via Tasky web application. While creating a project they can invite participants by indicating their e-mails. If there is already a user registered with the given e-mail, the system sends an invitation e-mail and mobile notification; otherwise, only an e-mail sent. Invited users can accept or decline the invitation. Creator of the project becomes the project manager. Project managers can assign a role to the project participants. Project managers can update the project by changing the status, name or description of the project. Project managers can add or remove participants to/from projects. Project managers can also delete the project.

Participants of the project can view project detail and project board in both platforms. Project detail contains name, description, participants and documents of the project. Project board contains tasks grouped by their status (To-do, active, resolved and closed).

Project managers can view the report of their projects via only web application. In the project report they can see the stats of tasks or team members.

### Task Management

Project managers and team members can create tasks under their projects via only Tasky web application. These tasks can have sub-tasks. The only difference between the task and sub-task is that sub-tasks have a root task.

These tasks can be assigned to a team member. Project managers can assign a task to anyone in the project except watchers. Team members can only assign tasks to themselves. Project managers and assignee of the task can update the task’s status, description, files. All participants of the project can view the detail of tasks. Project managers and team members can log work to the tasks. A task can be deleted by only a project manager. These functions are available in both platforms.

Project participants can view the task list of the project via both platforms. In this page they can filter or sort the tasks. Users also can view the last activities of their projects.

### Notifications

Project managers will receive e-mail and Mobile notifications when:

* A user accepts or declines their invitation.
* A task is updated.
* A work is logged into a task they created.
* A team member created a new task.
* A participant left the project.

Team members will receive e-mail and Mobile notifications when:

* A task is assigned to them.
* A work is logged into a task they created.
* A task that is assigned to them is updated or deleted.
* They are removed from a project.

## Nonfunctional Requirements

### Usability

* Users should be able to complete their transactions with maximum 5 clicks.
* The components of web and Mobile application such as titles, button names, section names etc. should be descriptive and understandable for people who knows English.

### Reliability

* The system should be available 100% of time.
* The system should validate the data that will be stored in the database and if there are errors, users should be notified by the system.

### Performance

* Response time of the system should be maximum 30 seconds for heavy functions like preparing a project report, for other functions it should be maximum 10 seconds.

### Supportability

* The system should be separated into packages/components to accomplish easy maintenance.

### Implementation

* The Mobile application should be implemented with React-native (JavaScript).
* Web application should be implemented with React (JavaScript).
* Web API’s that will be used in the Mobile and web application should be implemented with .NET 5 (C#).
* Database should be MSSQL.
* Visual Studio Code should be used for React and React-native. Visual Studio 2019 should be used for .NET 5.

### Interface

* There is no interface requirement in this project.

### Packaging

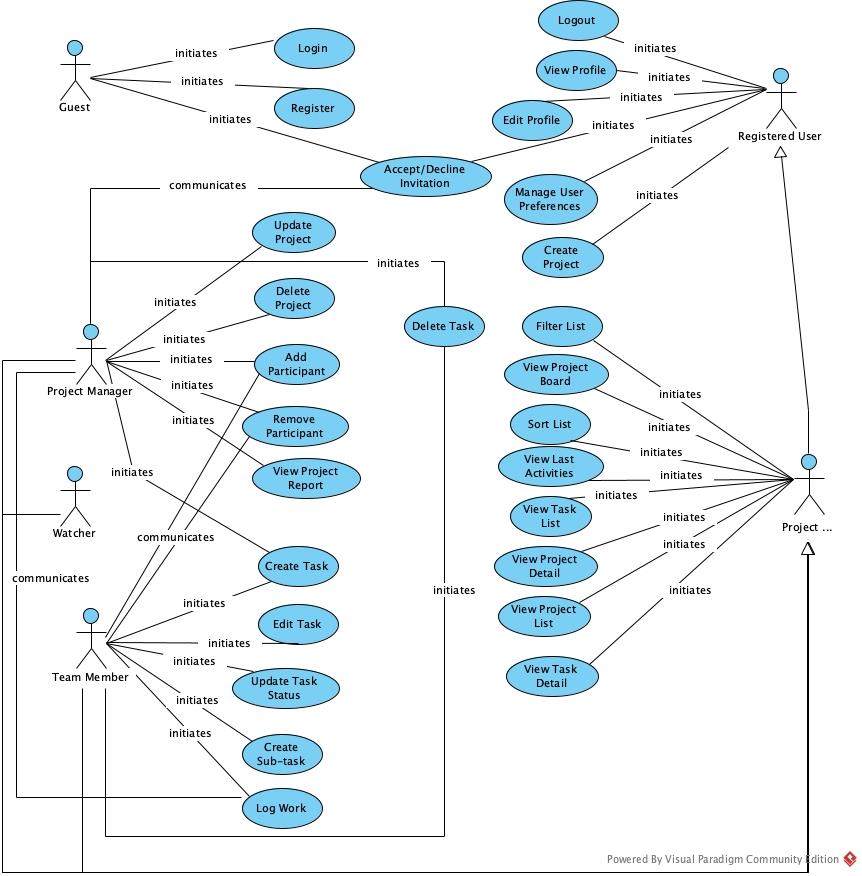
* Our web application does not require any installation. It should be run on any internet browser.
* Mobile application should be run on Android and iOS platforms by installing the application.

### Legal

* Mobile application should be signed with a signing key for android. For iOS, it should be signed with a certificate.

## Use Case Model

Use case model of the project that represents the functionalities of each role is given below.



**Figure 1.1 Use Case Model**

# Implementation, tests, experiments

## Implementation

### System Decomposition

## Tasky system is divided into three layers. First layer is interface. In this layer we have two subsystems.

1) **Tasky Web App:** Web application of the Tasky system which runs on internet browsers. All of the user interfaces and interactions in the Tasky web application belongs to this subsystem.

2) **Tasky Mobile App:** Mobile application of the Tasky system which runs on smart phones. All of the user interfaces and interactions in the Tasky mobile application belongs to this subsystem.

## Second layer of the Tasky system is application logic layer. This layer contains the server side controllers. Functions of this layer’s subsystems control and validate the data that is sent by user. There are 5 subsystems in this layer.

1)  **User Controller:** This subsystem is responsible of the user data. All operations related to user data such as profile and user preferences run on this subsystem. This subsystem also handles the login and register functions of the system. It creates a JWT token for the user and check its authorization in each API call.

2)  **Project Controller:** This subsystem handles the operations that is related to project.

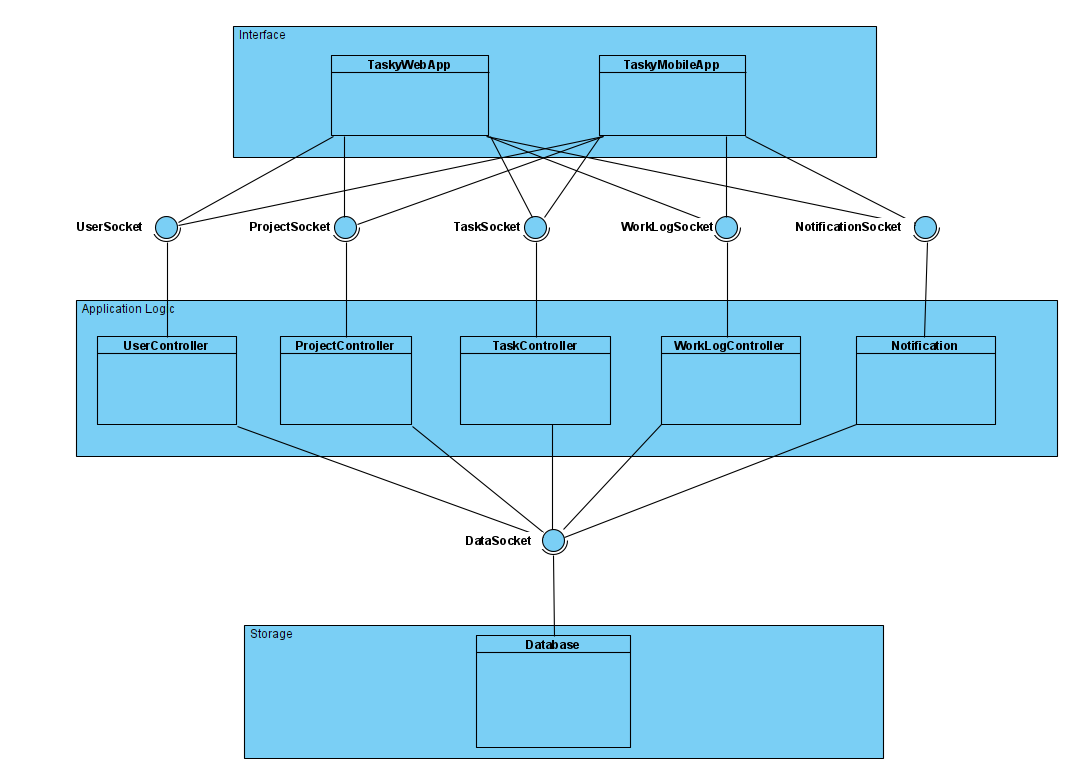
3)  **Task Controller:** This subsystem handles the operations that is related to task.

4)  **Work Log Controller:** This subsystem handles the operations that is related to work log.

5)  **Notification Controller:** This subsystem handles the sending notification to the users. Sending e-mail and mobile notification operations run on this subsystem.

Third and final layer of the system is storage layer. All the data of the system is stored and handled in this layer. This layer has a single subsystem.

1) **Database:** This subsystem stores all the data. It also makes insert, update and delete operations.



**Figure 1.2 System Decomposition**

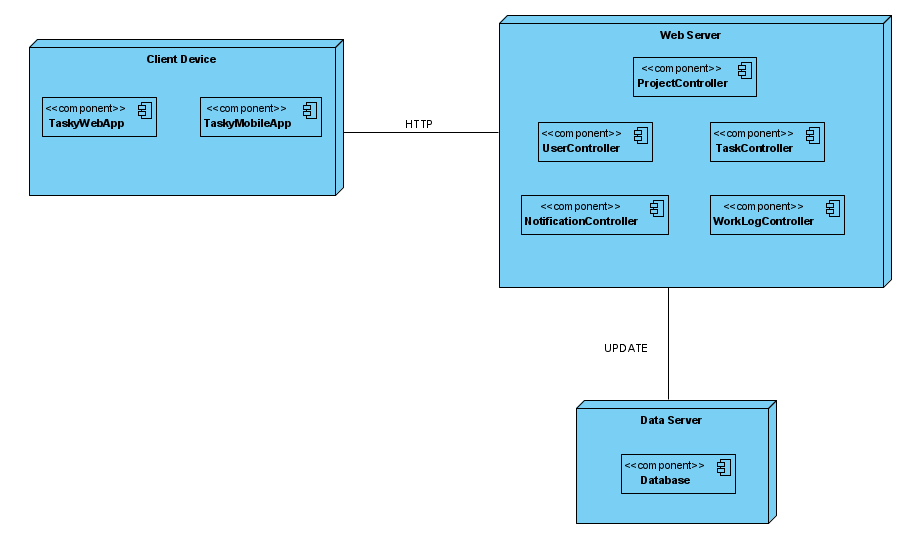
### Hardware-Software Mapping

## There are three hardware components in the system.

## First one is client device. Client device is the device that our web or mobile application run. It contains the components of Interface layer. It can be a computer or a smart phone.

## Second hardware component is web server. It contains the components of application logic layer. The operations are done in this hardware. And the

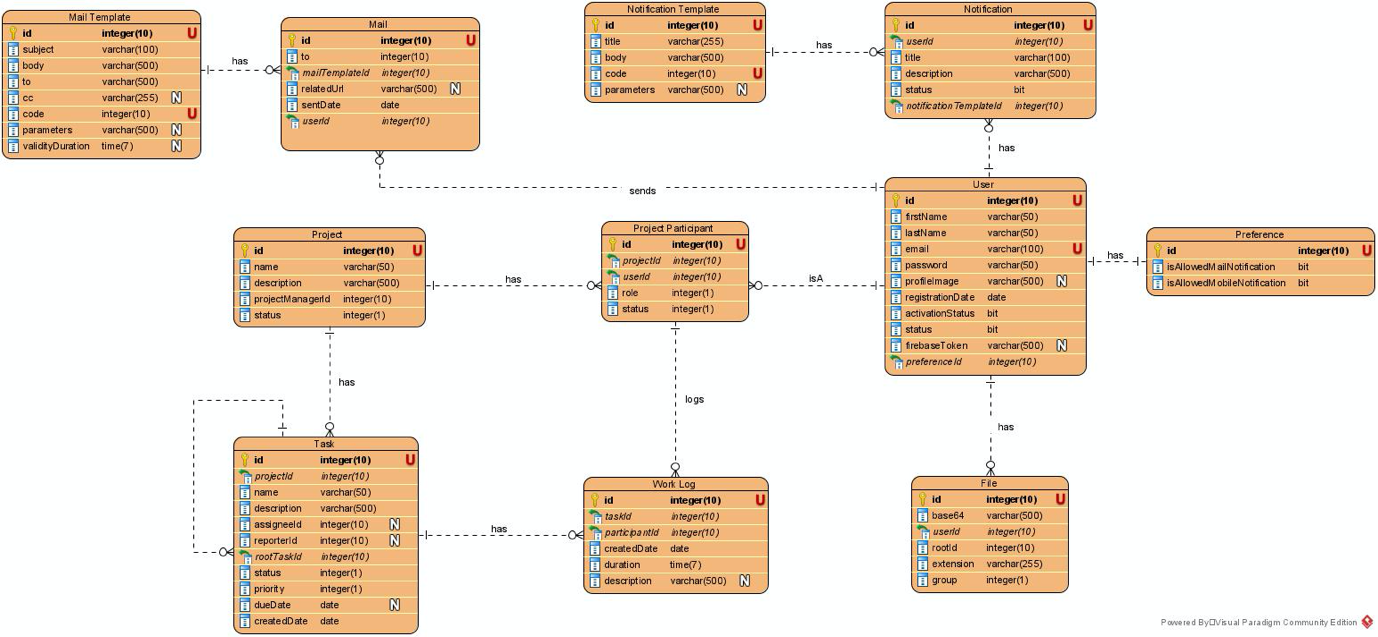
## Final hardware component is data server. It contains the database component. All of the dynamic data of the system is stored in this hardware component.



**Figure 1.3 Hardware-Software Mapping**

### Persistent Data Management

## Data of the Tasky system is stored in relational database which is run on the Azure server. System will keep the backup of the database in order to prevent any data loss. The password of the users will be kept as encrypted for avoiding the security problems. There will be 12 different tables in the database.



**Figure 1.4 Database of Tasky**

### Used Frameworks, Libraries and Templates

1. ***Web App Implementation***

## Tasky web application is implemented with React.js. React is a strong library that allows us implement web applications. There are a lot of npm packages used in the web project of Tasky. react-bootstrap, rsuite, semantic-ui-react, antd and material-ui are used for the UI components. All the charts in the tasky web application are taken from rechart package. Form validations in the tasky web application provided by Formik and yup packages.

1. ***Mobile App Implementation***

## Tasky mobile application is implemented with React-native. The advantage of the React-native is that we can run the application on both iOS and Android platforms with a single source code. This is the biggest reason of using react-native in Tasky. We also thought that using similar frameworks in mobile and web application will save time.

## We used a lot of npm packages in our mobile application. React-navigation package used for navigating between screens. Most of the UI elements of the Tasky mobile app are taken from react-native-paper and native-base packages. For the push notification feature of Tasky mobile app, we used Firebase. Firebase allows us to register iOS and Android apps to their system and send notifications. In the Tasky mobile Project we used react-native-firebase/app and react-native-firebase/messaging packages. Also, we took some of the UI elements from Antiqueruby material design template.

1. ***Web Service Implementation***

## Web services of the Tasky are implemented with .NET 5. Entity framework core package is used to connect the database with web services. We used database first approach in the Tasky. Database tables created first, then the classes that represents the tables and views in the database are implemented in the web service project. These web services are used in the web and mobile applications of the Tasky to make operations within the system. The data that is coming from client side of the system (web and mobile apps) is receives web services and processes to the database.

## Tests

Testing means software testing here. The OCL statements, if any, or tests designed and applied would be given in this part.

Pay special attention to this subsection, since “Test Design, Implementation, Running, Execution” is now one of the course outcomes, and will be graded.

Unit and/or integration test designs, fail/pass reports, and actions taken for fail cases are welcome.

## Experiments

If you have studied a research problem, worked on a learning algorithm for instance, you **must** present the experiments you conducted here. Also, compare and contrast your results with the existing approaches those presented in Chapter 2.

# conclusions and future work

Do not write “I have learned this this and this”, instead state the conclusion of your project, i.e., whether or not the success criteria have met, which design goals have been satisfied, the problems faced during the project design and/or implementation, etc.

Explain how this work can be extended in future.

# References

|  |  |
| --- | --- |
| [1] | IEEE, «IEEE Reference Guide,» IEEE, 2018. |
| [2] | J. A. Mitchell, A Guide to Citation, My London Publisher, 2017. |
| [3] | C. A. Petri, «Kommunikation mit automaten, (Communicating with automata)(in German),» Technical University Darmstadt, Germany, 1962. |
| [4] | «How to Cite Sources in MLA Citation Format-Mendeley,» [Çevrimiçi]. [Erişildi: 10 1 2021]. |