SMART - BRIDGE INTERNZ ORAGANIZATION

SAMHITHA
DEGREE COLLEGE

Team Members :-

E. Mani chandana (team leader) (200927109056)

1. M.Hima sindhuja (200927109066)

2. V.S.S.V.Suryaganesh (200927146107)

3. G.Nagendra babu (200927146103)

4. R.V.suresh (200927109076)

TODOAPP

INTRODUCTION:

ATODOAPPisasoftwareapplicationdesignedtohelpusersorganizeand managetheirtasksand to-dolists.It allowsuser’stocreate,prioritize,andtracktasks,setdeadlines ,andmarktasksascompleted.TODOAPPS arecommonlyusedforpersonalandprofessional productivitytostay organiseandensuretasksarecompletedontime.

Creating a TODOappinvolvesseveralsteps.Here'sahigh-leveloverviewofthe process:

1.Definethe Requirements:Determinethe featuresandfunctionalitiesyouwantin yourTODOapp, suchas taskcreation,task completion,priority levels, due dates,etc.

2.Choosea DevelopmentPlatform:Decidewhetheryouwanttodevelopa webapp, mobile app,ordesktopapp. Choosetheappropriateprogramming languagesandframeworksfor yourchosenplatform.

3.Designthe UserInterface: Createwireframesormockupsofyourapp'suser interfaceto planthelayout andinteractions

4.SetUpaDevelopmentEnvironment:Installthenecessarydevelop menttoolsandsoftwaretostartcoding yourTODOapp.

5.ImplementTaskManagement:Buildthe corefunctionalityfor creating,updating,anddeletingtasks.Storetaskdatainadatabase.

6.AddTaskPrioritizationandDue Dates:Implementfeaturestoassignprioritylevelsandduedatestot asks.

7.Implement TaskCompletion:Allowuserstomarktasksascompletedandup datetheir statusaccordingly.

8.UserAuthentication(Optional):Ifyouwantto adduseraccountsandsecuredata,implementuserauthentication.

9.Testing:ThoroughlytestyourTODOapp toensureit works correctlyandfix anybugs orissues.

10.Deployment:Choosea hostingplatformforyourapp (ifit'sa webapp)orsubmititto appstores(ifit'samobileapp).

11.MonitorandUpdate:Continuouslymonitor your app'sperformanceanduserfeedback. Regularlyupdateandimprovetheappbasedonuserneeds.

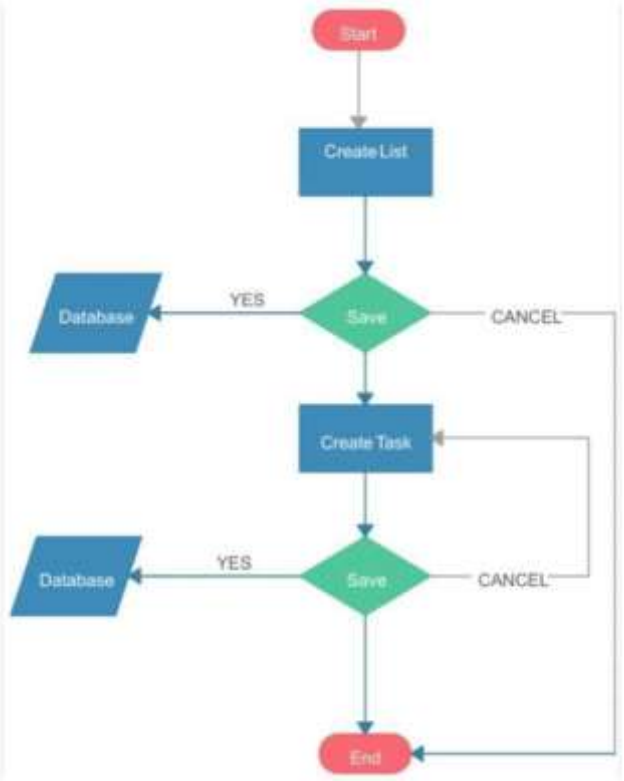
Prerequisites:

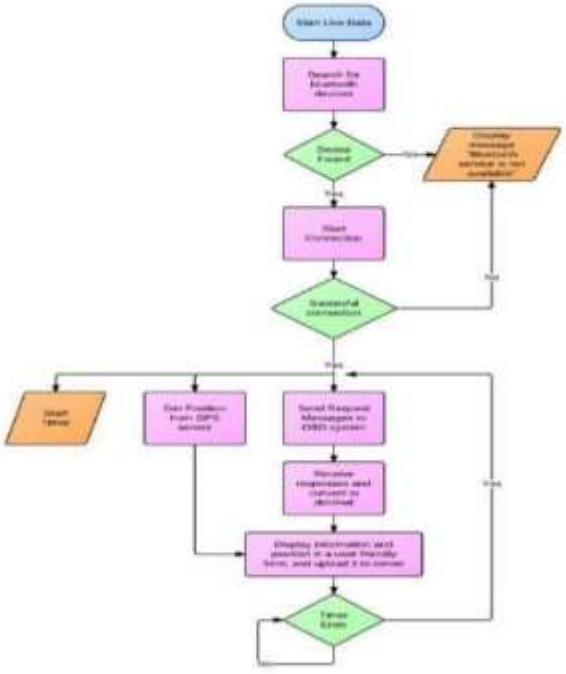
Beforewebeginwithourproject,wewouldlikeyoutoknowtheprerequisit esforthesamesoyoucanunderstandit.WehavemadeourprojectusingJav a programming, butAndroidevensupportsKotlinnow.

Pleaseconsiderthefollowingthingsthatyoumust havegoodhands with-

1. ObjectOrientedProgrammingConcepts
2. JavaProgramming /Kotlin
3. ExtensibleMarkupLanguage
4. AndroidStudio
5. AndroidVirtualMachince

FLOWCHART:-


Start
Create List
Database
YES
Save
CANCEL
Create Task
Database
YES
Save
CANCEL
End



PROJECTSTRUCTURE:-

SettinguptheprojectstructureforaTODOappinvolvesorganizingthecodeb aseandresourcesinasystematicway. Here'sabasicprojectstructureoutline:

TODO-App/

```
|-- app/
|   |-- src/
|   |   |--main/
|   |   |   |--java/
|   |   |   |   |--com.example.todoapp/
|   |   |   |   |   |--activities/
|   |   |   |   |   |   |--MainActivity.java
|   |   |   |   |   |--adapters/
|   |   |   |   |   |   |--TaskAdapter.java
|   |   |   |   |   |--data/
|   |   |   |   |   |   |--models/
|   |   |   |   |   |   |   |--Task.java
```

```
|   |   |   |   |   |   |--repositories/
|   |   |   |   |   |   |   |--TaskRepository.java
|   |   |   |   |   |--utils/
|   |   |   |   |   |   |--DateUtils.java
|   |   |   |   |   |--viewmodels/
|   |   |   |   |   |   |--TaskViewModel.java
|   |   |   |--res/
|   |   |   |   |--layout/
|   |   |   |   |   |--activity_main.xml
|   |   |   |   |--values/
|   |   |   |   |   |--strings.xml
|   |   |--test/
|   |   |   |--java/
|   |   |   |   |--com.example.todoapp/
|   |   |   |   |   |--TaskViewModelTest.java
|   |--build.gradle
|--gradle/
|   |--wrapper/
|   |   |--gradle-wrapper.properties
```

|--build.gradle

|--settings.gradle

|--gradle.properties

|--local.properties

|--gradlew

|--gradlew.bat

Designandimplementtheuserinterface:

Asan AIlanguagemodel,Icanprovidea generaloutlinefordesigning andimplementingthe userinterface(UI)ofaTODOapp. However,keepinmind that actualUIdesignandimplementationmightrequire additionaldetailsandconsiderationsbasedonyour specificrequirements andplatform(AndroidoriOS). Forthisexample,I'll outlinethestepsfor anAndroidappusingXMLlayout files:

1.*DesigntheUI*:

- Createa wireframeor sketchoftheapp's mainscreen, where userscanviewandmanagetheirtasks.

-Planthe layout, includingelementsliketasklist, taskitem,addtaskbutton,etc.

-Decide on thecolor schemeand typographyto maintaina consistent

visualstyle.2.*CreateXMLLayouts

*:

-Open the `activity_main.xml`layout fileinthe `res/layout`directory.

-DesignthemainscreenUIbyusingLinearLayout, RecyclerView,TextView,EditText,andButtonele mentsasneeded.

-Definethetask itemlayout(e.g., `task_item.xml`)todisplayindividualtasks in theRecyclerView.

3.*ImplementMainActivity*:

-Create`MainActivity.java`inthe`activities`pa ckage.

-InitializetheRecyclerViewanditsadapter.

-Set uptheRecyclerViewwithaLayoutManager todisplaythelistoftasks.

- Implementthelogictofetchtasksfromthe repositoryandpassthemtotheRecyclerView adapter.

4.*ImplementTaskAdapter*:

- Create`TaskAdapter.java`inthe`adapters`pa ckage.
- ExtendRecyclerView.Adapterandimplement requiredmethods.
- Inflate the`task_item.xml`layoutin `onCreateViewHolder`.
- Bindtaskdatatotheviewsin `onBindViewHolder`.
- Implementitemclick listenerstohandletask interactions.4.*Handl

eAddTask*:

- Add an EditTextandButtonforadding new taskstothelayout.
- Implementlogictohandle the"AddTask" buttonclick.

- Whenthebuttonisclicked,retrievethetask from the EditText, create a new Task object, andadd ittotherepository.

6.*ImplementTaskModel*:

- Createthe`Task.java`classinthe `data/models`package.
- Definepropertiesforatask,suchasID,title, description,duedate,etc.

6.*ImplementTaskRepository*:

- Create`TaskRepository.java`inthe `data/repositories`package.
- Implementmethodsto handleCRUDoperationsfortasks (e.g.,addTask,getAllTasks,deleteTask).

6.*ImplementTaskViewModel(Optional)*:

- Create`TaskViewModel.java`inthe `viewmodels` package(ifusingMVVM architecture).
- ImplementtheViewModel to handledatabetweenMainActivityandTaskRe pository.

9.*ApplyStylesandThemes*:

- Definestylesandthemesinthe`res/values`directoryto m

aintainaconsistentlookandfeelacrosstheapp.

10.*TestandIterate*:

-Testtheapp thoroughly toensureallfunctionalitiesworkasexpected.

-Iterate on the design and implementation basedon userfeedbackand tofixanybugs.

Remember,thisisa basicoutline,and youmayneedtomodifyor extendit dependingonyourapp'scomplexityandrequirements. Additionally,it'sessential tofollowAndroiddevelopmentguidelines andbestpracticeswhiledesigningand

implementing theUI

Implementtaskmanagement functionality:

```
<!doctype html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>TO DO List</title>
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-GLhlTQ8iRABdZLl6O3oVMWSktQOp6b7In1Zl3/Jr59b6EGGoI1aFkw7cmDA6j6gD" crossorigin="anonymous">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.3.0/css/all.min.css" integrity="sha512-SzlrxWUlpfuzQ+pcUCosxcglQRNAq/DZjVsC0lE40xsADsfeQoEypE+enwcOiGjk/bSuGGKHEyjSoQ1zVisanQ==" crossorigin="anonymous" referrerpolicy="no-referrer" />
<link href="style.css" rel="stylesheet">
</head>
<body>

<div class="container">
<header id="main-header" class="bg-warning text-white p-4 mb-3">
<div class="row">
<div class="col-md-12">
<h1 id="header-title">ToDo List
<i class="fa fa-pencil" style="float:right"></i>
</h1>
</div>
</div>
</header>
<div class="card">
<div class="card-body">
<div class="input-group mb-3">
<input type="text"
class="form-control" id="inputField" placeholder="Enter item to do...">
<button type="button" class="btn btn-primary" id="addToDo">
```

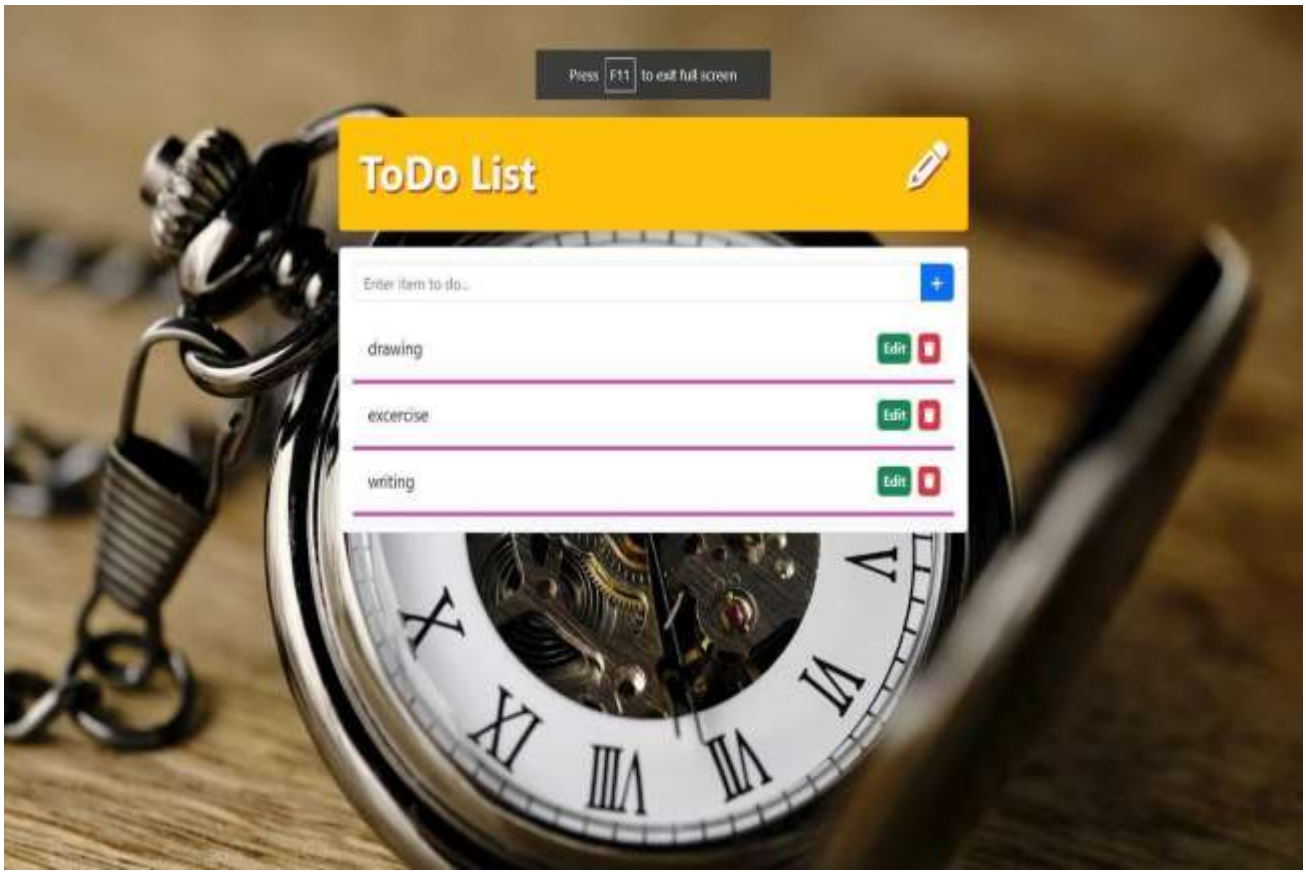
```
<i class="fa fa-add"></i></h1>
</button>
</div>
<div class="to-dos" id="toDoContainer">
</div>
                                        </div>
                                        </div>
</div>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js"
integrity="sha384-w76AqPfDkMBDXo30jS1Sgez6pr3x5MlQ1ZAGC+nuZB+EYdgRZgiwxhTBTkF7CXvN"
crossorigin="anonymous"></script>
<script src="script.js"></script>
                                        </body>
</html>
*{
padding: 0;
margin: 0;
box-sizing: border-box;
font-family: cursive sans-serif ;
}
body{
height:100vh;
background-image:url(images/clock-3179167.jpg); background-repeat: no-repeat;
background-size:cover; overflow: hidden;
}
#main-header{ border-radius: 5px;
}
#header-title{ font-size: 3rem;
font-weight: 700;
text-shadow: 2px 2px 2px #b41c1c;
}
.container{ width:50%;
margin-top: 7em;
}
@media screen and (max-width: 800px) {
```

```
.container{ width:70%;
}
}
@media screen and (max-width: 650px) {
.container{ width:80%;
}
}
@media screen and (max-width: 500px) {
.container{ width:99%; margin-top: 5em;
}
}

.item{
display: flex;
flex-direction: row; padding: 1rem ;
border-bottom:4px solid rgb(211, 61, 191)
   ;
}

.content{ display:1 1 0%; width:90%;
}
.item .content .text { font-size: 1.125rem;
width: 100%;
border-color: transparent;
}
.item .content .text:hover { cursor:context-
   menu;
}
.item .actions { display: flex;
}
.item .actions button {
margin-left: 0.5rem; font-weight: 700; font-
   size:0.9rem;
```

```
padding:2px 6px;
}

var add = document.getElementById('addToDo'); var input =
document.getElementById('inputField');
var toDoContainer =
document.getElementById('toDoContainer');

add.addEventListener('click',addItem);
input.addEventListener('keypress',function(e){
if(e.key=="Enter"){ addItem();
}
});
function addItem(e){

const item_value  = input.value;
const item = document.createElement('div');
item.classList.add('item');

const item_content = document.createElement('div');
item_content.classList.add('content');

item.appendChild(item_content);

const input_item = document.createElement('input');
input_item.classList.add('text');
input_item.type = 'text'; input_item.value = item_value;
input_item.setAttribute('readonly', 'readonly');
input_item.addEventListener('dblclick', function(){
input_item.style.textDecoration = "line-through";
})
item_content.appendChild(input_item);

const item_action = document.createElement('div');
item_action.classList.add('actions');

const edit_item = document.createElement('button');
edit_item.classList.add('edit','btn','btn-success');
edit_item.type="button";
edit_item.innerText = 'Edit';
```

```
const delete_item = document.createElement('button');
delete_item.classList.add('delete','btn','btn-danger','fa','fa-trash');

item_action.appendChild(edit_item);
item_action.appendChild(delete_item);

item.appendChild(item_action);

toDoContainer.appendChild(item);


input.value = ''; edit_item.addEventListener('click', (e) => {
if (edit_item.innerText.toLowerCase() == "edit") {
edit_item.innerText = "Save";
input_item.removeAttribute("readonly"); input_item.focus();
} else {
edit_item.innerText = "Edit"; input_item.setAttribute("readonly",
"readonly");
}
});

delete_item.addEventListener('click', (e) => {
toDoContainer.removeChild(item);
});
}
```

Output


Press F11 to exit full screen
ToDo List
Enter item to do...
drawing
Edit
excercise
Edit
writing
Edit


THANKS
FOR WATCHING