# TASK MANAGEMENT SYSTEM

**PROJECT REPORT**

**(2021 Regulation)**

***Submitted by***

**C.MANEESH KUMAR (RA2211047010094)**

**B.VISHWANATH (RA2211047010082)**

**II Year/ IIIrd Semester**

**Academic Year: 2023 -2024**

***Under the Guidance of***

**Dr. SUMATHY G**

**Assistant Professor, Department of Computational Intelligence**

***In partial satisfaction of the requirements for the degree of***

**BACHELOR OF TECHNOLOGY**

**In**

**ARTIFICIAL INTELLIGENCE**



**SCHOOL OF COMPUTING**

**COLLEGE OF ENGINEERING AND TECHNOLOGY SRM INSTITUTE OF SCIENCE AND TECHNOLOGY KATTANKULATHUR - 603203**

# NOV / DEC 2023

**Logo, company name

Description automatically generated**

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

**BONAFIDE CERTIFICATE**

Certified that this Course Project Report titled **“TASK MANAGEMENT SYSTEM”** is the bonafide work done by **B.VISHWANATH[RA2211047010082],C.MANEESH KUMAR[RA2211047010094]** of II Year/ IIIrd Sem B.Tech (AI) who carried out under my supervision for the course **21CSC101T** - **Object Oriented Design and Programming**. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

|  |  |
| --- | --- |
| **SIGNATURE**  Faculty In-Charge  **Dr. Sumathy G**  Assistant Professor  Department of Computational Intelligence  SRM Institute of Science and Technology  Kattankulathur Campus, Chennai | **HEAD OF THE DEPARTMENT**  **Dr. R Annie Uthra**  Professor and Head,  Department of Computational Intelligence,  SRM Institute of Science and Technology  Kattankulathur Campus, Chennai |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **CONTENTS** | **PAGE NO** |
|  | **Abstract** |  |
| **1** | **Objective** | **6** |
| 2 | Problem Statement | **7** |
| 3 | Modules of Project | **8** |
| 4 | UML Diagrams |  |
|  | 3.1. Use case Diagram | **9** |
|  | 3.2. Class Diagram | **10** |
|  | 3.3. Sequence Diagram | **11** |
|  | 3.4. Collaboration Diagram | **12** |
|  | 3.5. State Chart Diagram | **13** |
|  | 3.6. Activity Diagram | **14** |
|  | 3.7. Package Diagram | **15** |
|  | 3.8. Component Diagram | **16** |
|  | 3.9. Deployment Diagram | **17** |
| 5 | Code/Output Screenshots | **18 – 30** |
| 6 | Conclusion | **31** |
|  | References | **32** |

# Abstract:

# In an era marked by constant multitasking and increasing demands on our time, efficient Taskmanagement is paramount for personal and professional success. The "Comprehensive Task Management System" is a robust software solution designed to simplify and optimize the management of tasks. This system encompasses a suite of essential features, including task creation, viewing, deletion, updating, and task searching.

# Users can effortlessly add tasks, providing task details such as names, descriptions, due dates, and priorities. They can categorize tasks, attach files, and establish dependencies to create intricate task structures. Viewing tasks is made easy through an intuitive interface, enabling users to filter and sort tasks based on various criteria. Task updates are a breeze, allowing users to modify task details or change due dates to adapt to evolving requirements. The system also facilitates task deletion, aiding in maintaining an organized task list.

# Moreover, the system empowers users with a search functionality, making it effortless to locate specific tasks within a vast repository. This "Comprehensive Task Management System" is an indispensable tool for individuals and teams looking to enhance productivity, meet deadlines, and maintain clarity in an increasingly complex world.

# 1.OBJECTIVE:

# The primary objective of the Task Management System is to provide a robust and user-friendly platform for efficient and organized task management. This system aims to enhance productivity, reduce stress, and improve time management by offering a comprehensive set of features, including task creation, viewing, deletion, updating, and task searching.

# 1. Task Creation: The system's foremost objective is to facilitate the easy and quick addition of tasks. Users can enter task details, such as task name, description, due date, priority, and tags, ensuring that no task is overlooked or forgotten.

# 2. Task Viewing: The system allows users to view their tasks in a structured and organized manner. Users can filter, sort, and categorize tasks, providing a clear overview of their responsibilities.

# 3.Task Update: Task details are not static; they often change over time. The system's objective is to empower users to modify task information, ensuring that it remains accurate and aligned with evolving requirements.

# 4. Task Deletion: To maintain a clutter-free task list, the system enables users to delete completed or unnecessary tasks easily.

# 5. Task Searching: Searching for specific tasks among a growing list is a core objective. The system facilitates quick and efficient task retrieval based on keywords, due dates, or other criteria.

# In summary, the Task Management System's objective is to empower users with the tools necessary for effective task management, ultimately helping individuals and teams meet their goals, stay organized, and reduce the risk of missed deadlines and overlooked responsibilities. This system strives to enhance overall productivity and time utilization.

# 2.PROBLEM STATEMENT:

# Many individuals and organizations face the challenge of efficiently managing their tasks, resulting in missed deadlines, disorganization, and decreased productivity. The need for a comprehensive Task Management System has become increasingly evident. This system aims to address the problem of task mismanagement by providing a solution that enables users to add, view, update, delete, and search tasks effectively. The lack of a centralized and user-friendly task management tool hinders productivity, collaboration, and goal achievement. The primary problem is the absence of a structured approach to handle tasks, which often leads to chaos and missed opportunities. This Task Management System intends to alleviate these issues and enhance task management efficiency, making it a valuable tool for individuals and organizations seeking to optimize their time and resources.

# MODULES OF PROJECT:

# A Task Management System typically consists of various modules to handle its features effectively. Here are the essential modules for a task management system that include features like adding tasks, viewing tasks, deleting tasks, updating tasks, and more:

# 1. User Management Module :

# - User registration and authentication.

# - User profiles and preferences.

# 2. Task Creation Module :

# - Adding new tasks with details like name, description, due date, priority, and tags.

# - Attachment of files or documents to tasks.

# - Task dependency management (e.g., linking related tasks).

# 3. Task Viewing and Organization Module :

# - Task dashboard or list view.

# - Sorting and filtering tasks by attributes (due date, priority, category, etc.).

# - Task categorization and tagging for better organization.

# 4. Task Update Module :

# - Editing and updating task details.

# - Changing due dates, priorities, and other attributes.

# - Managing task statuses (e.g., marking tasks as completed).

# 5. Task Deletion Module :

# - Deleting tasks that are no longer relevant.

# - Archiving completed tasks for historical reference.

# 6. Search and Retrieval Module :

# - Advanced search functionality for finding specific tasks.

# - Search by keywords, due dates, tags, or other criteria.

# Use -case diagram:

# WhatsApp Image 2023-11-03 at 08.54.22_c6b17657

# In the use case diagram for a Task Management System, the primary actorsare the User and potentially an Administrator with higher privileges. The main use cases include "Add Task," where the User initiates the addition of a new task by providing details like title and due date. "Remove Task" allows the User to delete a selected task from the system after confirmation. "Update Task" enables the User to modify task details, such as title or description. Lastly, "Delete Task" involves permanently removing a task, with confirmation steps. Associations connect actors to their respective use cases, illustrating the interactions. This diagram encapsulates the essential functionalities of the system from a user's perspective, providing a concise representation of task management actions.

# Class diagram:

# WhatsApp Image 2023-11-03 at 08.55.11_105e415e

# In the class diagram for a Task Management System, key classes include "Task" with attributes such as title, description, and due date. Each task is associated with the "User" class, representing individuals interacting with the system. Operations like "Add Task," "Remove Task," "Update Task," and "Delete Task" are encapsulated within the "TaskManager" class, responsible for managing tasks. The class diagram also denotes the relationships, indicating that a user can be associated with multiple tasks, establishing a one-to-many relationship. Additionally, the "TaskManager" class is likely to interact with a data storage class, representing a database or file system to persistently store task information. This diagram provides a structural overview of the classes and their associations, highlighting the organization of classes and their roles in implementing the specified features within the Task Management System.

# sequence diagram:

# WhatsApp Image 2023-11-03 at 08.56.15_fd5ffa6d

# 

# In a sequence diagram for a Task Management System, the focus is on depicting the chronological order of interactions between objects. For the "Add Task" sequence, the User sends a request to add a task, prompting the system to gather task details before adding it to the task list. In the "Remove Task" sequence, the User initiates the removal process, leading to a confirmation step before the system removes the selected task. The "Update Task" sequence begins with the User triggering an update request, the system displaying the task list, and allowing the User to modify details, followed by the system updating the task. In the "Delete Task" sequence, the User requests task deletion, the system confirms, and then permanently removes the task. These sequences provide a dynamic view of how users and the system interact over time, offering insights into the flow of tasks within the Task Management System.

# Collaboration diagram:

# WhatsApp Image 2023-11-03 at 08.57.46_0c79a0ca

# In a collaboration diagram for a Task Management System, the emphasis is on illustrating the structural organization of objects and how they collaborate to achieve specific functionalities. For the "Add Task" collaboration, the User object interacts with the Task Management System object, which in turn collaborates with a Task object to encapsulate and store the new task details. In the "Remove Task" collaboration, the User interacts with the Task Management System, which collaborates with the Task object to identify and remove the selected task. The "Update Task" collaboration involves interactions between the User, Task Management System, and Task objects to modify and update task details. Lastly, in the "Delete Task" collaboration, the User communicates with the Task Management System, which collaborates with the Task object to confirm and permanently delete the specified task. This diagram provides a visual representation of how objects in the system collaborate to execute the key functionalities of task addition, removal, update, and deletion in the Task Management System.

# State chart diagram:

# WhatsApp Image 2023-11-03 at 09.03.16_6a790bc6

# In a State Chart Diagram for a Task Management System, the primary focus is on representing the different states that an object (such as a task or the system itself) can transition through in response to events. For the "Add Task" feature, the initial state may be "Idle," transitioning to "Collecting Details" when the user initiates the task addition. Upon receiving the necessary details, it moves to the "Updating Task List" state before returning to the "Idle" state. Similarly, the "Remove Task" feature could involve states like "Idle," "Selecting Task," "Confirmation," and finally, "Task Removed." The "Update Task" feature may have states such as "Idle," "Selecting Task," "Modifying Details," and "Task Updated." Lastly, the "Delete Task" feature might encompass states like "Idle," "Selecting Task," "Confirmation," and "Task Deleted." State Chart Diagrams visually illustrate the life cycle and transitions between states, providing a clear depiction of the system's behavior for each task management feature.

# Activity diagram:

# WhatsApp Image 2023-11-03 at 09.06.04_c3f63c69

# In an activity diagram for a Task Management System, the emphasis is on illustrating the workflow and activities involved in the key features such as adding, removing, updating, and deleting tasks. For the "Add Task" activity, the diagram would show the sequential steps, starting with the User initiating the task addition, followed by the system prompting for task details, and ultimately concluding with the task being added to the system. The "Remove Task" activity demonstrates the user selecting a task for removal, the system confirming the action, and then executing the removal. Similarly, the "Update Task" activity outlines the steps of selecting a task, modifying details, and the system updating the task. The "Delete Task" activity encompasses the user's request, system confirmation, and the subsequent permanent removal of the task. These activities, visually represented in the diagram, provide a clear and concise overview of the dynamic processes involved in the Task Management System.

# Package diagram:

# WhatsApp Image 2023-11-03 at 09.07.47_3079a90e

# In a package diagram for a Task Management System, the emphasis is on organizing and structuring the system's components. The system can be divided into packages such as "User Interface," "Task Management," and "Database." Within the "Task Management" package, there are sub-packages for each feature, including "Add Task," "Remove Task," "Update Task," and "Delete Task." Each of these sub-packages encapsulates the relevant classes and components responsible for the specific functionality. For instance, the "Add Task" sub-package may contain classes related to capturing user input and handling the addition of tasks. This hierarchical structure provides a clear representation of how the system is modularized, making it easier to manage, maintain, and understand. Additionally, it allows for better organization of classes and components, supporting the scalability and extensibility of the Task Management System.

# Component diagram:

# WhatsApp Image 2023-11-03 at 09.09.18_f922b2ba

# In a component diagram for a Task Management System, the system's structural organization is illustrated, emphasizing the key components and their interactions. The primary components include "User Interface," "Task Manager," and "Database." The "User Interface" component facilitates interactions between the User and the system for features like "Add Task," "Remove Task," "Update Task," and "Delete Task." The "Task Manager" component serves as the core logic, managing the execution of these features by processing user requests and coordinating with the "Database" component. The "Database" component stores and retrieves task data, ensuring persistence. Associations between components represent the flow of data and control. This diagram offers a high-level view of how different parts of the Task Management System collaborate to provide functionality, aiding developers and stakeholders in understanding the system's architecture.

# Deployment diagram:

# 

# WhatsApp Image 2023-11-03 at 09.12.14_0dd2396c

# In a Deployment Diagram for a Task Management System, the emphasis is on illustrating the physical deployment of components across nodes or hardware. The system architecture involves a User Device (like a computer or mobile device) interacting with a centralized Task Management Server. For the "Add Task" feature, the User Device communicates with the server, which processes the request and updates the task database. In the "Remove Task" scenario, the User Device sends a request to the server, which then interacts with the task database to remove the specified task. Similarly, for "Update Task," the User Device communicates modification details to the server, which subsequently updates the task database. The "Delete Task" feature follows a similar pattern, with the User Device initiating a request handled by the server. This deployment model clarifies the distribution of responsibilities between the user interface and the server, providing an overview of the physical arrangement of components in the Task Management System.

# PROGRAM :

# #include <iostream>

# #include <fstream>

# #include<string>

# #include <cstdlib>

# using namespace std;

# int ID;

# struct todo {

# int id;

# string task;

# };

# void addtodo() {

# system("cls");

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

# cout<<"\t\* To Do List Application \*"<<endl;

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl<<endl<<endl;

# todo todo;

# cout << "\n\tEnter new task: ";

# cin.get();

# getline(cin, todo.task);

# ID++;

# ofstream write;

# write.open("todo.txt", ios::app);

# write << "\n" << ID;

# write << "\n" << todo.task ;

# write.close();

# write.open("id.txt");

# write << ID;

# write.close();

# char ch;

# cout<<"\n\tDo you want to add more task? y/n: ";

# cin>> ch;

# if(ch == 'y'){

# addtodo();

# }

# 

# else{

# cout << "\n\tTask has been added successfully";

# return;

# }

# }

# void print(todo s) {

# cout << "\n\tID is : " << s.id;

# cout << "\n\tTask is : " << s.task;

# }

# void readData() {

# system("cls");

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

# cout<<"\t\* To Do List Application \*"<<endl;

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl<<endl<<endl;

# todo todo;

# ifstream read;

# read.open("todo.txt");

# cout << "\n\t------------------Your current Tasks in the list--------------------";

# while (!read.eof()) {

# read >> todo.id;

# read.ignore();

# getline(read, todo.task);

# print(todo);

# }

# read.close();

# }

# int searchData() {

# system("cls");

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

# cout<<"\t\* To Do List Application \*"<<endl;

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl<<endl<<endl;

# int id;

# cout << "\n\tEnter task id: ";

# cin >> id;

# todo todo;

# ifstream read;

# read.open("todo.txt");

# while (!read.eof()) {

# read >> todo.id;

# read.ignore();

# getline(read, todo.task);

# if (todo.id == id) {

# print(todo);

# return id;

# }

# }

# }

# void deleteData() {

# system("cls");

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

# cout<<"\t\* To Do List Application \*"<<endl;

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl<<endl<<endl;

# int id = searchData();

# cout << "\n\tDo you want to delete this task (y/n) : ";

# char choice;

# cin >> choice;

# if (choice == 'y') {

# todo todo;

# ofstream tempFile;

# tempFile.open("temp.txt");

# ifstream read;

# read.open("todo.txt");

# while (!read.eof()) {

# read >> todo.id;

# read.ignore();

# getline(read, todo.task);

# if (todo.id != id) {

# tempFile << "\n" << todo.id;

# tempFile << "\n" << todo.task;

# }

# }

# read.close();

# tempFile.close();

# remove("todo.txt");

# rename("temp.txt", "todo.txt");

# cout << "\n\tTask deleted successfuly";

# }

# else {

# cout << "\n\tRecord not deleted";

# }

# }

# void updateData() {

# system("cls");

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

# cout<<"\t\* To Do List Application \*"<<endl;

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl<<endl<<endl;

# int id = searchData();

# cout << "\n\n\tYou want to update this task (y/n) : ";

# char choice;

# cin >> choice;

# if (choice == 'y') {

# todo newData;

# cout << "\n\tEnter todo task : ";

# cin.get();

# getline(cin, newData.task);

# todo todo;

# ofstream tempFile;

# tempFile.open("temp.txt");

# ifstream read;

# read.open("todo.txt");

# while (!read.eof()) {

# read >> todo.id;

# read.ignore();

# getline(read, todo.task);

# if (todo.id != id) {

# tempFile << "\n" << todo.id;

# tempFile << "\n" << todo.task;

# }

# else {

# tempFile << "\n"<< todo.id;

# tempFile << "\n"<< newData.task;

# }

# }

# read.close();

# tempFile.close();

# remove("todo.txt");

# rename("temp.txt", "todo.txt");

# cout << "\n\tTask updated successfuly";

# }

# else {

# cout << "\n\tRecord not deleted";

# }

# }

# int main()

# {

# system("cls");

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

# cout<<"\t\* To Do List Application \*"<<endl;

# cout<<"\t\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl<<endl<<endl<<endl;

# ifstream read;

# read.open("id.txt");

# if (!read.fail()) {

# read >> ID;

# }

# else {

# ID = 0;

# }

# read.close();

# while (true) {

# cout<<endl;

# cout << "\n\t1.Add Task";

# cout << "\n\t2.Display Task";

# cout << "\n\t3.Search Task";

# cout << "\n\t4.Delete Task";

# cout << "\n\t5.Update Task";

# int choice;

# cout << "\n\n\tEnter choice : ";

# cin >> choice;

# switch (choice) {

# case 1:

# addtodo();

# break;

# case 2:

# readData();

# break;

# case 3:

# searchData();

# break;

# case 4:

# deleteData();

# break;

# case 5:

# updateData();

# break;

# }

# }

# }

# Program :

# 

# 

# 

# 

# 

# 

# 

# 

# Output :

# WhatsApp Image 2023-11-03 at 11.35.44_002ddb8c

# WhatsApp Image 2023-11-03 at 11.37.43_67ed63a0

# WhatsApp Image 2023-11-03 at 11.38.56_8c6ef40a

# WhatsApp Image 2023-11-03 at 11.40.02_975a34d2

# WhatsApp Image 2023-11-03 at 11.40.48_281d52de

# WhatsApp Image 2023-11-03 at 11.41.40_21feb879

# Conclusion;

# "In conclusion, the Task Management System project has successfully addressed the need for efficient and organized task management. The system was designed to provide users with a user-friendly and feature-rich platform for managing tasks, including adding, viewing, updating, searching, and deleting tasks. Throughout the project, several essential objectives were achieved:

# 1. Efficient Task Management: The system provides a practical solution for managing tasks, improving productivity, and ensuring that tasks are organized and tracked effectively.

# 2.Comprehensive Features: The system includes a wide range of features, including task creation, viewing, updating, searching, and deletion, as well as user collaboration, notifications, and reporting.

# 3.Data Persistence: Tasks and user data are stored persistently, ensuring that task lists and user preferences are retained across sessions.

# 4.Security: The system implements access control and security measures to protect user data and maintain confidentiality.

# In the end, the Task Management System project has delivered a valuable tool for individuals and organizations seeking to improve their task management processes, stay organized, and enhance productivity. It is an essential resource for keeping track of tasks, meeting deadlines, and achieving goals effectively."

# Refference :

# S. McKenna, "Organisational Complexity and Perceptions of Task", Task Management: An International Journal, vol. 3, no. 2, pp. 53-64, 2013.

# A. Aleshin, "Time and Risk Management of International Projects", International Journal of Project Management, no. 19, pp. 207-222, 2014.

# M. Hopkinson, "Using Task Priority Models" in , Briefing:Kluwer's Task Management, vol. 40, pp. 4-8, 2014.

# M. Czerwinski, E. Cutrell and E. Horvitz, "Instant Messaging and Interruption: Influence of Task Type on Performance", Proceedings of OZCHI 2000, pp. 356-361, 2010.

# A. K. Dey, G. D. Abowd and A. Wood, "CyberDesk: A Framework for Providing Self-Integrating Tasks", Proceedings of the International Conference on Intelligent User Interfaces (IUI '98), pp. 47-54.