Programming III

BMEVIIIAB00

To-Do List Application

Muhammad Ibrahim Shoeb
BSc Computer Science Engineering, BME
OZLVV3

1. Description of the Task

The proposed project is a Task Management System implemented as a desktop application using Java. This application will help users organize and manage their daily tasks efficiently while implementing all required programming concepts from the course.

Core Features

Task Management

- Create new tasks with titles, descriptions, and due dates
- Edit existing tasks to update their information
- Delete tasks that are no longer needed
- Mark tasks as complete or incomplete
- Set priority levels (High, Medium, Low) for tasks
- Add detailed descriptions to tasks

Task Organization

- Categorize tasks into different groups (Work, Personal, School, etc.)
- Sort tasks by various criteria:
  - Due date
  - Priority level
  - Completion status
  - Category
- Filter tasks based on their status, priority, or due date

Data Storage

- Save tasks to local storage using Java serialization
- Load previously saved tasks when the application starts
- Automatic saving of changes to prevent data loss

User Interface

- Clean and intuitive Swing-based graphical interface
- Table view showing task details (using JTable)
- Drop-down menus for selecting priorities and categories (using JComboBox)
- Menu bar for accessing various application features

Technical Implementation

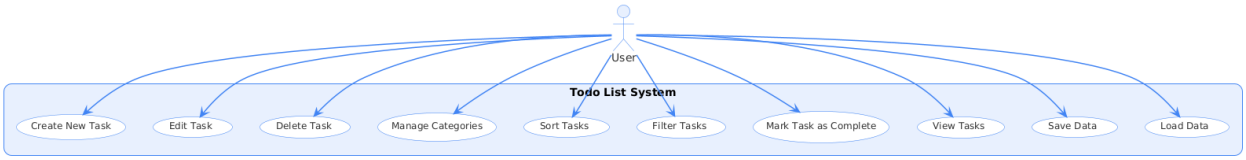
The application will utilize the following required components:

1. **Swing-based GUI**
   - JTable for displaying the list of tasks
   - JComboBox for selecting task priorities
   - Menu bar for application navigation
   - Dialog windows for task creation/editing

2. **Collections Framework**
   - ArrayList for storing tasks
   - HashMap for managing categories
   - Custom sorting implementations
3. **File I/O with Serialization**
   - Save task data to files
   - Load task data from files
   - Automatic backup functionality
4. **Unit Tests**
   - Test task creation and modification
   - Test data persistence operations
   - Test sorting and filtering functionality

2. Use-Case Diagrams and Descriptions

Use Case Diagram



Use Case Descriptions

1. Create New Task

- **Actor**: User
- **Description**: User creates a new task in the system
- **Basic Flow**:
  1. User clicks “New Task” button
  2. System displays task creation form
  3. User enters task details (title, description, due date, priority)
  4. User clicks “Save”
  5. System adds task to the list
- **Alternative Flow**: User cancels task creation

2. Edit Task

- **Actor**: User
- **Description**: User modifies an existing task
- **Basic Flow**:
  1. User selects a task
  2. User clicks “Edit” button
  3. System displays task details in editable form
  4. User modifies details
  5. System updates task information

3. View Tasks

- **Actor**: User
- **Description**: User views list of tasks
- **Basic Flow**:
    1. User opens application
    2. System displays task list
    3. User can sort/filter tasks
    4. User can select different views

4. Manage Categories

- **Actor**: User
- **Description**: User manages task categories
- **Basic Flow**:
    1. User selects category management
    2. System shows category list
    3. User can add/edit/delete categories
    4. System updates category structure

3. Brief Implementation Plan

The application will be implemented using the following approach:

Phase 1: Core Structure

1. Set up basic project structure in IntelliJ
2. Implement core classes (Task, Category)
3. Create basic data structures using Collections Framework
4. Implement file I/O with serialization

Phase 2: User Interface

1. Design main application window
2. Create task table view using JTable
3. Implement task creation/editing forms
4. Add menu bar and basic controls

Phase 3: Features

1. Add sorting and filtering functionality
2. Implement category management
3. Create file saving/loading system
4. Add additional user interface features

Phase 4: Testing

1. Write unit tests using JUnit
2. Test all core functionality
3. Verify data persistence