8 Task 3.2

Question 8. Draw an implementation diagram for Task Assignment module

Solution

8.1 Theoretical basis: Component diagram

UML Component diagrams are used in modeling the physical aspects of object-oriented systems that are used for visualizing, specifying, and documenting component-based systems and also for constructing executable systems through forward and reverse engineering. Component diagrams are essentially class diagrams that focus on a system's components that often used to model the static implementation view of a system.
A component diagram breaks down the actual system under development into various high levels of functionality. Each component is responsible for one clear aim within the entire system and only interacts with other essential elements on a need-to-know basis.
A component diagram has some common concepts:

- **Component:** represents a modular part of a system that encapsulates its contents and whose manifestation is replaceable within its environment. In UML 2.x, a component is drawn as a rectangle with optional compartments stacked vertically.
- **Interface:** 2 types:
  - **Required interface:** represents an interface that the component requires, drawn as a socket (half-circle).
  - **Provided interface:** represents an interface that the component provides, drawn as a lollipop (full-circle).
- **Port:** often used to help expose required and provided interfaces of a component, represented as a square placed along the edge of the component rectangle.
- **Relationship:** includes association, composition, aggregation, constraint, links and dependency. Their meaning and representation in UML are the same with those in class diagram.

8.2 Component diagram of Task assignment module

8.2.1 Component diagram

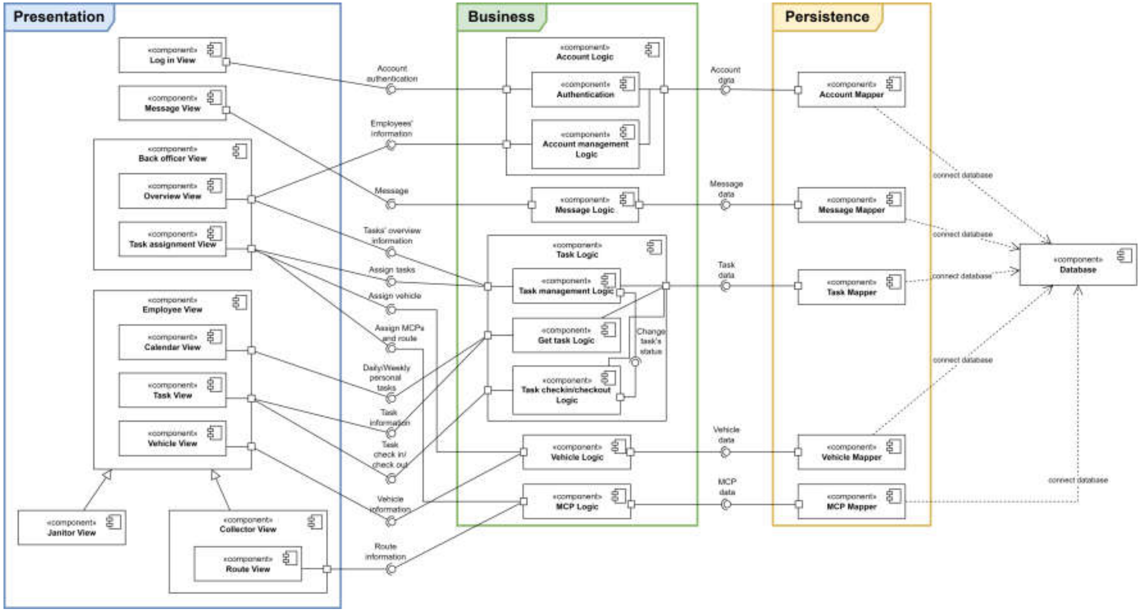

Figure 3: Component diagram of Task assignment module of UWC 2.0

Follow this link for a PNG file with better resolution, or this link for the diagram file.

8.2.2 Description and explanation

The component diagram is divided into 4 different layers corresponding to layered architecture that we chose in task 3.1 (we did not include Service layer in the diagram since it is just a layer separated from the Business layer to include some shared and external API for the components in Business layer), with each layer has the following components:

1. **Presentation layer:**
   - **Log in View:** displays the log in UI to user, uses interface Account authentication from component Account Logic (internal component Authentication) to perform its functionality.
   - **Message View:** displays the message UI to user, uses interface Message exposed by component Message Logic.
   - **Back officer View:**
     - **Overview View:** displays the overview UI for back officer by using interfaces Employees' information provided by component Account Logic (internal component Account Management Logic) and Tasks' overview information provided by component Task Logic (internal component Task management Logic).
     - **Task assignment View:** displays the task assignment view for back officer, requires interfaces Assign tasks, Assign vehicle and Assign MCPs and route from components Task Logic (internal component Task management Logic), Vehicle logic and MCP Logic respectively.
   - **Employee View:**

   - **Calendar View:** displays calendar UI for collector and janitor, uses interfaces Daily/Weekly personal tasks from component Task Logic (internal component Get task Logic).
   - **Task View:** displays individual task and its details to collector and janitor by using interfaces Task information and Task check in/check out from component Task Logic (internal component Get task Logic and Task check in/check out Logic respectively).
   - **Vehicle View:** displays vehicle information to collector and janitor by using interfaces Vehicle information from component Vehicle Logic.
   - Component Employee View is inherited by component Janitor View, which displays UI for janitor, and component Collector View, which displays UI for collector. Component Collector View has another internal component Route View displaying the route UI for collector by using interface Route information from component MCP Logic.

2. **Business layer:** consists of 5 major components representing 5 modules of the Business layer we have discussed in task 3.1 which are Account Logic, Message Logic, Task Logic, Vehicle Logic and MCP Logic, each of which requires the data object from its respective data mapper component to perform its business logic.

3. **Persistence layer:** consists of 5 data mappers which are Account Mapper, Message Mapper, Task Mapper, Vehicle Mapper and MCP Mapper, all of which establish a connection to component Database in order to retrieve and perform query on the data.

4. **Data layer:** consists of Database providing data to each data mapper in the Persistence layer.