#### **Household Installations Service Simulations – (HISS)**

HISS is a simulation which will allow trainee project managers to practice managing projects offering a complete household installation service for fixed furniture such as cupboards, wardrobes, kitchen fixtures. You are part of a team developing this simulation package, using an “incremental" development model. The package will eventually include time management and project task analysis. Your team is looking at cost management only. Requirements for this stage of the development have been established and specified in an interface. Your task is to produce a prototype of this simulation, by implementing the specified interface and providing some trainee interaction. Some of your code will be used in later stages. It is very important that your prototype is designed in a way which will make it easy to maintain and to add further functionality.

* 1. **System Specification**

A trainee project manager needs to assemble a project team from the hireable staff available. An installation project consists of a number of tasks, each to be performed by one member of their team. Depending on the task, the system allocates a member of staff to do the task, and works out the outcome. At the start of a project, an initial budget is paid into the project account. The trainee project manager will need to hire staff members to the project team and pay them a retainer, but is not allowed to hire staff if there is not enough money in the project account. Staff may be hired at any time during the project. The trainee project manager must manage their team and the tasks so that the project account does not end up below zero.

Currently, there are three types of staff: Installer, Consultant, Planner and three types of tasks: Design, Installation, or Maintenance. This table shows which type of staff can to be used to do each type of task.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Design** | **Installation** | **Maintenance** |
| **Planner** | yes | no | no |
| **Installer** | no | yes | Only, if trained |
| **Consultant** | yes | yes | yes |

Each staff member has a unique name, experience, a retainer and an hourly rate. Experience is recorded as a number from 1 to 10. A retainer is paid to ensure staff are available during the project, even if not required all of the time. Installers have a retainer of £200 and an hourly rate of £20. They can do maintenance tasks, but only if trained to work with electricity. Consultants set their own retainer. If they are CORGI certified, their hourly rate is £40, else their hourly rate is £30. Planners have a retainer of £300 and the make of fixture they specialise in. Their hourly rate is calculated by multiplying their experience by 15.

Staff are in one of the following states:

* available – is in the collection of hireable staff, available but not yet hired for the team
* working – has been hired for the team and is available to work on jobs
* on leave – is in the team but is not available for jobs; can rejoin the team

Each job has a unique sequential job number starting from 1000, a type (Design, Installation, Maintenance), a location, the number of hours it should take to complete, a difficulty level (a number between 1-10) and a penalty. To make money, the manager must select jobs by specifying their job number. The application will then choose the first available staff member who can do the job. The result of doing a job will be one these::

* "Job completed by….“ –add job cost to project account, staff member goes “on leave”
* “Job not completed, as no staff available” – delay costs money, so deduct job penalty from account
* “Job not completed due staff inexperience” - deduct job penalty from account
* "No such job" - no further action taken

HISS **always** displays the state of the project account after processing each job.

The cost of a job depends on the number of hours and the hourly rate of the member of staff chosen to do the job. At this stage of the project, there is **NO** need to find the cheapest member of the team who can perform the job. For a job to be completed, its difficulty level must be less than the chosen staff member’s experience. When a job has been completed, the staff member who was selected is entitled to a rest, so their state is set to “on leave” and they cannot be used for further jobs until they re-join the team. If a job is completed, the team gets paid and the project account is increased. If not completed, a penalty must be paid from the account.

* 1. **Functional Requirements**

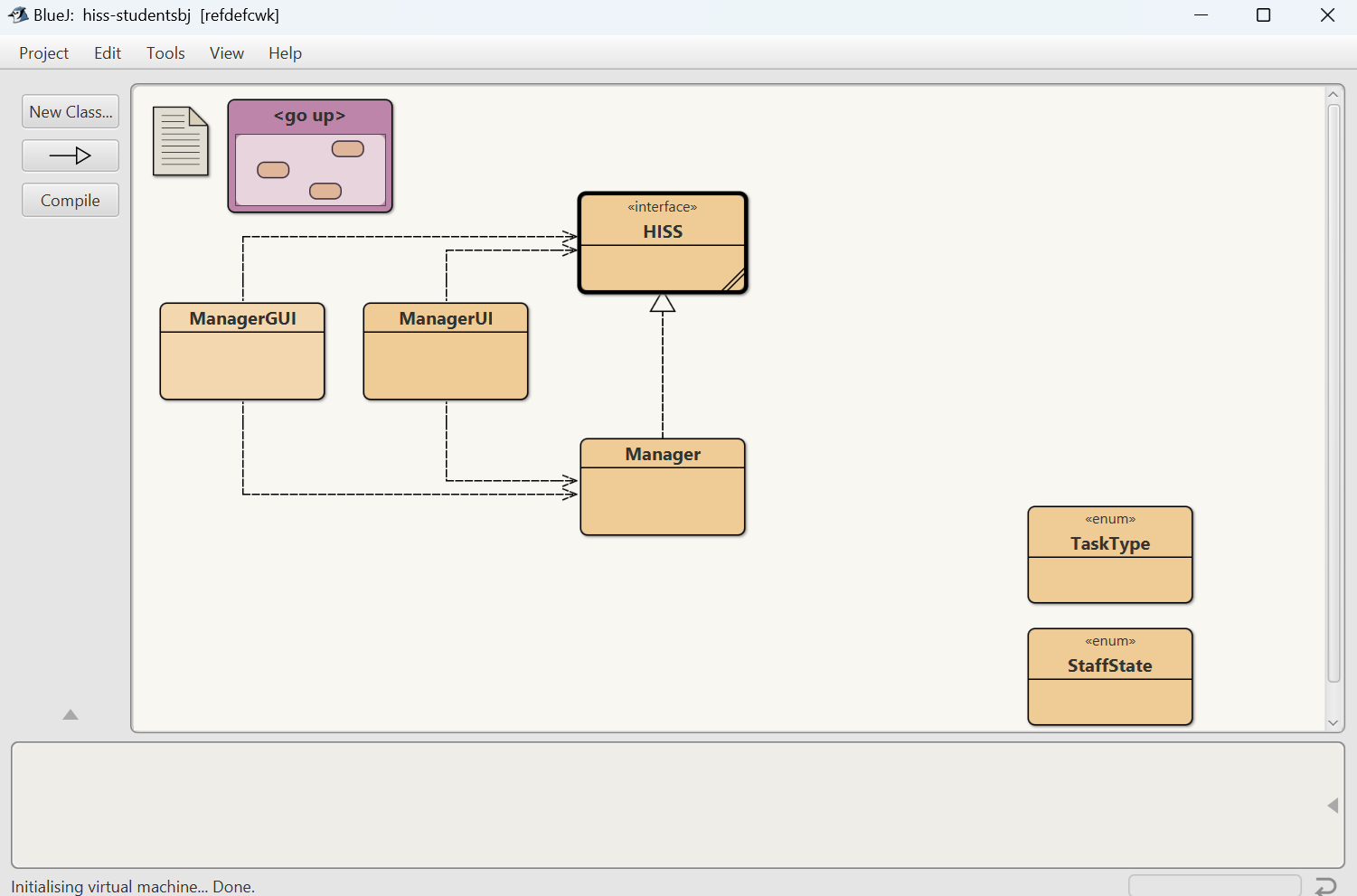
When a trainee manager starts a simulation project, they will be asked for their name and the project budget. The system will create an instance of the simulation and automatically load all the sample data (see Appendix A) of staff and jobs into appropriate variables/collections.   
  
The trainee should be able to create a project team by hiring some of the available staff. On hiring a staff member, appropriate deductions should be made from project account. If there is not enough money in the project account or the staff not available, the staff cannot be hired. Appropriate messages about the outcomes should be displayed.

Once the trainee has some staff in their team, they can perform project jobs. In this prototype, the trainee will be required to provide the job number. The system will then select the first appropriate staff to perform the job. The result of the job will be processed as described in the specification above and a suitable message displayed. In this version, jobs may be performed more than once, and the trainee may view jobs.

The trainee may hire more staff at any point in the simulation. The trainee may request to see: the state of the simulation, the state of the project account, all staff available for hire, all the staff in the team, all jobs, details of one staff member

Most commercial projects separate the user "front end" which handles interaction with the user from the "business" end which provides the functionality i.e. performs the processing. These functional requirements have been specified in the interface HISS. The methods in this interface have parameters which are only basic types or String. The interface makes no assumptions about supplier classes in your design.

Classes provided in hiss-studentsNB



**Appendix A**

**Staff**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Name** | **Experience** | **Retainer** | **Corgi** | **Trained** | **Make** | **HourlyRate** |
| Planner | **Amir** | **2** | 300 |  |  | Homebase | 2\*15=30 |
| Consultant | **Bela** | **2** | **100** | **false** |  |  | 30 |
| Consultant | **Ceri** | **4** | **250** | **true** |  |  | 40 |
| Installer | **Dana** | **2** | 200 |  | no |  | 20 |
| Installer | **Eli** | **7** | 200 |  | yes |  | 20 |
| Planner | **Firat** | **6** | 300 |  |  | Hygena | 6\*15 = 90 |
| Installer | **Gani** | **2** | 200 |  | yes |  | 20 |
| Consultant | **Hui** | **8** | **450** | **true** |  |  | 40 |
| Planner | **Jaga** | **4** | 300 |  |  | Homebase | 4 \* 15 = 60 |

**Tasks**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task No** | **Type** | **Location** | **Experience Required** | **Hours** | **Penalty** |
| 1000 | Design | Kitchen | 3 | 10 | 200 |
| 1001 | Maintenance | Lounge | 3 | 20 | 150 |
| 1002 | Installation | Kitchen | 3 | 30 | 100 |
| 1003 | Design | Bathroom | 9 | 25 | 250 |
| 1004 | Installation | Lounge | 7 | 15 | 350 |
| 1005 | Maintenance | Kitchen | 8 | 35 | 300 |
| 1006 | Maintenance | Bathroom | 5 | 20 | 400 |
| 1007 | Installation | Bathroom | 1 | 5 | 120 |
| 1008 | Design | Kitchen | 1 | 8 | 175 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Design** | **Installation** | **Maintenance** |
| **Planner** | yes | no | no |
| **Installer** | no | yes | Only, if trained |
| **Consultant** | yes | yes | yes |