**Use Case Scenarios**

**Employee**

|  |  |
| --- | --- |
| *Scenario name* | EmployeeLogin |
| *Participating actor*  *instances* | bob:Employee |
| *Flow of events* | 1. Bob wants to perform some function related to his goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager takes submitted form and performs form validation. GoalManager attempts to validate Bob’s credentials by hashing the provided password, querying its database for an entry matching Bob’s username, and comparing the associated hash value with the hashed password submitted by Bob. GoalManager serves the default webpage for validated Employees to Bob’s browser. 2. Bob is able to view his goals, set new goals for himself, and update progress on his goals. |

|  |  |
| --- | --- |
| *Scenario name* | EmployeeInvalidLogin |
| *Participating actor*  *instances* | bob:Employee |
| *Flow of events* | 1. Bob wants to perform some function related to his goals, so he opens a browser, navigates to the network resource hosting GoalManager, but provides invalid credentials to the login webpage form.      1. GoalManager takes submitted form and performs form validation. GoalManager attempts to validate Bob’s credentials by hashing the provided password, querying its database for an entry matching Bob’s username, and comparing the associated hash value with the hashed password submitted by Bob. GoalManager returns a password-mismatch error to the login webpage. 2. Bob receives an invalid login error on the login webpage. |

|  |  |
| --- | --- |
| *Scenario name* | EmployeeViewGoals |
| *Participating actor*  *instances* | bob:Employee |
| *Flow of events* | 1. Bob wants to check on his progress towards his quarterly and yearly goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Bob’s credentials, presenting the default webpage for Employees. 2. Bob navigates to the button that activates the “Goal Progress Report” function. 3. GoalManager queries goal data from database and presents it to Bob through the “Goal Progress Report” webpage. 4. Bob is able to view the personal goals he set for himself, as well any goals assigned by his Supervisor. |

|  |  |
| --- | --- |
| *Scenario name* | EmployeeCreateGoal |
| *Participating actor*  *instances* | bob:Employee |
| *Flow of events* | 1. Bob wants to set a new goal for himself, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Bob’s credentials, presenting the default webpage for Employees. 2. Bob navigates to the button that activates the “Set New Goal” function. 3. GoalManager presents the webpage interface for setting new Employee goals to Bob. 4. Bob populates the goal’s title, time class, content, and category drop-down field in the webpage form. Bob submits the form by clicking the Submit button. 5. GoalManager performs form validation on entries. Bob’s submission is valid, and GoalManager writes new a new goal to the database. GoalManager returns Bob back to the default Employee webpage. |

|  |  |
| --- | --- |
| *Scenario name* | EmployeeSetGoalFailure |
| *Participating actor*  *instances* | bob:Employee |
| *Flow of events* | 1. Bob wants to set a new goal for himself, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Bob’s credentials, presenting the default webpage for Employees. 2. Bob navigates to the button that activates the “Set New Goal” page. 3. GoalManager presents the webpage interface for setting new Employee goals to Bob. 4. Bob populates the goal’s title, time class, content, and category drop-down field in the webpage form. Bob submits the form by clicking the Submit button, however some form data is invalid. 5. GoalManager performs form validation on entries. Bob’s submission is invalid, and GoalManager returns Bob back to the webpage associated with the “Set New Goal” function. |

|  |  |
| --- | --- |
| *Scenario name* | EmployeeUpdateGoal |
| *Participating actor*  *instances* | bob:Employee |
| *Flow of events* | 1. Bob wants to update his progress towards his quarterly and yearly goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Bob’s credentials, presenting the default webpage for Employees. 2. Bob navigates to the button that activates the “Goal Progress Report” function. 3. GoalManager queries goal data from database and presents it to Bob through the webpage. 4. Bob selects one of his personal or assigned goals to update from the “Goal Progress Report” webpage. 5. GoalManager serves dynamic content on the “Goal Progress Report” webpage with a form requesting updated progress information. 6. Bob enters valid data into the “Goal Progress Form” page’s dynamic webpage and presses “Submit”. 7. GoalManager performs form validation on Bob’s submitted form data. Passing that, GoalManager writes updated progress for the selected goal to its database. GoalManager returns Bob back to the default Employee webpage. |

**Supervisor Scenarios**

|  |  |
| --- | --- |
| *Scenario name* | SupervisorLogin |
| *Participating actor*  *instances* | steve:Supervisor |
| *Flow of events* | 1. Steve wants to perform some function related to his goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager takes submitted form and performs form validation. GoalManager attempts to validate Steve’s credentials by hashing the provided password, querying its database for an entry matching Steve’s username, and comparing the associated hash value with the hashed password submitted by Steve. GoalManager serves the default webpage for validated Supervisors to Steves’s browser. 2. Steve is able to view his goals, set new goals for himself, and update progress on his goals. |

|  |  |
| --- | --- |
| *Scenario name* | SupervisorInvalidLogin |
| *Participating actor*  *instances* | Steve:Supervisor |
| *Flow of events* | 1. Steve wants to perform some function related to his goals, so he opens a browser, navigates to the network resource hosting GoalManager, but provides invalid credentials to the login webpage form.      1. GoalManager takes submitted form and performs form validation. GoalManager attempts to validate Bob’s credentials by hashing the provided password, querying its database for an entry matching Steve’s username, and comparing the associated hash value with the hashed password submitted by Steve. GoalManager returns a password-mismatch error to the login webpage. 2. Steve receives an invalid login error on the login webpage. |

|  |  |
| --- | --- |
| *Scenario name* | SupervisorViewGoals |
| *Participating actor*  *instances* | steve:Supervisor |
| *Flow of events* | 1. Steve wants to check on his progress towards his quarterly and yearly goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Steve’s credentials, presenting the default webpage for Employees. 2. Steve navigates to the button that activates the “Goal Progress Report” function. 3. GoalManager queries goal data from database and presents it to Steve through the “Goal Progress Report” webpage. 4. Steve is able to view the personal goals he set for himself, as well as every goal for each of his employees. |

|  |  |
| --- | --- |
| *Scenario name* | SupervisorCreateGoal |
| *Participating actor*  *instances* | steve:Supervisor |
| *Flow of events* | 1. Steve wants to set a new goal for himself, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Steve’s credentials, presenting the default webpage for Supervisors. 2. Steve navigates to the button that activates the “Set New Goal” function. 3. GoalManager presents the webpage interface for setting new Supervisor goals to Steve. 4. Steve populates the goal’s title, time class, content, and category drop-down field in the webpage form. Steve submits the form by clicking the Submit button. 5. GoalManager performs form validation on entries. Steve’s submission is valid, and GoalManager writes a new goal to the database. GoalManager returns Steve back to the default Supervisor webpage. |

|  |  |
| --- | --- |
| *Scenario name* | SupervisorSetGoalFailure |
| *Participating actor*  *instances* | steve:Supervisor |
| *Flow of events* | 1. Steve wants to set a new goal for himself, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Steve’s credentials, presenting the default webpage for Supervisors. 2. Steve navigates to the button that activates the “Set New Goal” page. 3. GoalManager presents the webpage interface for setting new Supervisor goals to Steve. 4. Steve populates the goal’s title, time class, content, and category drop-down field in the webpage form. Steve submits the form by clicking the Submit button, however some form data is invalid. 5. GoalManager performs form validation on entries. Steve’s submission is invalid, and GoalManager returns Steve back to the webpage associated with the “Set New Goal” function. |

|  |  |
| --- | --- |
| *Scenario name* | SuprevisorUpdateGoal |
| *Participating actor*  *instances* | steve:Supervisor |
| *Flow of events* | 1. Steve wants to update his progress towards his quarterly and yearly goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Steve’s credentials, presenting the default webpage for Supervisors. 2. Steve navigates to the button that activates the “Goal Progress Report” function. 3. GoalManager queries goal data from database and presents it to Steve through the webpage. 4. Steve selects one of his personal or assigned goals to update from the “Goal Progress Report” webpage. 5. GoalManager serves dynamic content on the “Goal Progress Report” webpage with a form requesting updated progress information. 6. Steve enters valid data into the “Goal Progress Form” page’s dynamic webpage and presses “Submit”. 7. GoalManager performs form validation on Steve’s submitted form data. Passing that, GoalManager writes updated progress for the selected goal to its database. GoalManager returns Steve back to the default Supervisor webpage. |

|  |  |
| --- | --- |
| *Scenario name* | SuprevisorViewReport |
| *Participating actor*  *instances* | steve:Supervisor |
| *Flow of events* | 1. Steve wants to view Employee records, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates Steve’s credentials, presenting the default webpage for Supervisors. 2. Steve navigates to the button that activates the “View Report” function. 3. GoalManager queries a list of Employee names from the database and presents it to Steve through the webpage. 4. Steve selects one of the Employees to view from the “View Report” webpage. 5. GoalManager serves content on the “View Report” webpage with a form displaying the Employee’s records. 6. Steve exits out of the “View Report Form” webpage and presses “Close”. 7. GoalManager returns Steve back to the default Supervisor webpage. |

**Administrator Scenarios**

|  |  |
| --- | --- |
| *Scenario name* | AdministratorLogin |
| *Participating actor*  *Instances* | John:Administrator |
| *Flow of events* | 1. John wants to perform some function related to his goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager takes submitted form and performs form validation. GoalManager attempts to validate John’s credentials by hashing the provided password, querying its database for an entry matching John’s username, and comparing the associated hash value with the hashed password submitted by John. GoalManager serves the default webpage for validated Employees to John’s browser. 2. John is able to view his goals, set new goals for himself, and update progress on his goals. |

|  |  |
| --- | --- |
| *Scenario name* | AdministratorInvalidLogin |
| *Participating actor*  *Instances* | John:Administrator |
| *Flow of events* | 1. John wants to perform some function related to his goals, so he opens a browser, navigates to the network resource hosting GoalManager, but provides invalid credentials to the login webpage form.      1. GoalManager takes submitted form and performs form validation. GoalManager attempts to validate John’s credentials by hashing the provided password, querying its database for an entry matching John’s username, and comparing the associated hash value with the hashed password submitted by John. GoalManager returns a password-mismatch error to the login webpage. 2. John receives an invalid login error on the login webpage. |

|  |  |
| --- | --- |
| *Scenario name* | AdministratorViewGoals |
| *Participating actor*  *instances* | John:Adminstrator |
| *Flow of events* | 1. John wants to check on his progress towards his quarterly and yearly goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates John’s credentials, presenting the default webpage for Employees. 2. John navigates to the button that activates the “Goal Progress Report” function. 3. GoalManager queries goal data from database and presents it to John through the “Goal Progress Report” webpage. 4. John is able to view the personal goals he set for himself, as well any goals assigned by his Supervisor. |

|  |  |
| --- | --- |
| *Scenario name* | AdministratorCreateGoal |
| *Participating actor*  *instances* | John:Adminstrator |
| *Flow of events* | 1. John wants to set a new goal for himself, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates John’s credentials, presenting the default webpage for Employees. 2. John navigates to the button that activates the “Set New Goal” function. 3. GoalManager presents the webpage interface for setting new Employee goals to John. 4. John populates the goal’s title, time class, content, and category drop-down field in the webpage form. John submits the form by clicking the Submit button. 5. GoalManager performs form validation on entries. John’s submission is valid, and GoalManager writes new a new goal to the database. GoalManager returns John back to the default Employee webpage. |

|  |  |
| --- | --- |
| *Scenario name* | AdministratorSetGoalFailure |
| *Participating actor*  *instances* | John:Adminstrator |
| *Flow of events* | 1. John wants to set a new goal for himself, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates John’s credentials, presenting the default webpage for Employees. 2. John navigates to the button that activates the “Set New Goal” page. 3. GoalManager presents the webpage interface for setting new Employee goals to John. 4. John populates the goal’s title, time class, content, and category drop-down field in the webpage form. John submits the form by clicking the Submit button, however some form data is invalid. 5. GoalManager performs form validation on entries. John’s submission is invalid, and GoalManager returns John back to the webpage associated with the “Set New Goal” function. |

|  |  |
| --- | --- |
| *Scenario name* | AdminstratorUpdateGoal |
| *Participating actor*  *instances* | John:Administrator |
| *Flow of events* | 1. John wants to update his progress towards his quarterly and yearly goals, so he opens a browser, navigates to the network resource hosting GoalManager, and logs in with his credentials.      1. GoalManager performs form validation and validates John’s credentials, presenting the default webpage for Employees. 2. John navigates to the button that activates the “Goal Progress Report” function. 3. GoalManager queries goal data from database and presents it to John through the webpage. 4. John selects one of his personal or assigned goals to update from the “Goal Progress Report” webpage. 5. GoalManager serves dynamic content on the “Goal Progress Report” webpage with a form requesting updated progress information. 6. John enters valid data into the “Goal Progress Form” page’s dynamic webpage and presses “Submit”. 7. GoalManager performs form validation on John’s submitted form data. Passing that, GoalManager writes updated progress for the selected goal to its database. GoalManager returns John back to the default Employee webpage. |

Scenario name: Add Employee

*Participating actors*: John:Administrator

*Flow of events*: 1. The Administrator presses the Add Employee Button.  
**2. The system will prompt a form for the Administrator to fill out about the Employee.**3. The Administrator fills out the form and presses the Submit button.  
**4. The Employee is added to the system and the Administrator is prompted back to the main menu.**

*Entry condition*: Administrator must be logged in

*Exit condition*: Administrator Submits an employee to the system.

*Security requirements*:

**2.3 USE CASE DESCRIPTIONS**

*Use case name*: Remove Employee

*Participating actors*: Administrator

*Flow of events*: 1. The Administrator selects the Remove Employee button on the Main Menu.

**2. The System then prompts a list of Employees.**

3. The Administrator then selects an Employee to Remove and Presses the Remove button.

**4. The System then removes the Employee from the Database and prompts to the main menu.**

*Entry condition*: Administrator must be logged in

*Exit condition*: Administrator must select an employee to remove.

*Security requirements*: