**Part 1: Analyze the requirements and create test scenarios**.

1. **Features of Salary Hero App**

The feature chain represents the logical flow of user interactions with the earned wage withdrawal application. This chain demonstrates the dependencies between features and the typical user journey through the application.

* 1. **User Authentication** with Phone Numbermust succeed before any downstream features.
     1. Entry point to the application.
     2. Authentication gateway for all subsequent features.
     3. Log in by other methods.
     4. Prerequisite for accessing any personal wage information.
  2. **Dashboard/Home** pulls data from the back-end, so it must run every time the dashboard is loaded.
     1. Depends on successful login
     2. Greeting & summary cards
        + - Displays a day worked bar for use in calculations.
          - Provides information necessary for making withdrawal decisions (display prorated daily rate × days worked) e.g. 12th day of the month.

Available balance for withdrawal.

Total withdrawals this pay cycle.

* + 1. View latest Top 3 Transaction History.
  1. **Withdraw Earned Wage** depends on the current available balance for withdrawal.
     1. Depends on successful login and viewing Dashboard/Home.
     2. Input withdrawal amount that allows employees to request a portion or all their available earned wage.
     3. Initiates the transaction process.
  2. **Transaction Confirmation & Slip** is triggered by a successful withdrawal.
     1. View withdraws details
     2. Input OTP to confirm transaction
     3. View details of a successful transaction.
  3. **Transaction History** always reflects both historical and just-submitted withdrawals.
     1. Filter: by Period.
     2. View transaction information with collapse.
     3. View transaction information with expand to see details.
  4. **Session Management** applies globally across all screens.
     1. Idle timeout
     2. Manual logout

**QUESTIONS**

* **User Authentication:** Should there be any additional authentication steps (PIN, SMS OTP, or password)?
* **Withdrawal limits:** Is there a minimum/maximum withdrawal amount or frequency constraint?
* **Business-day logic:** How is “12th working day” determined (including holidays/weekends)?
* **Transaction statuses:** What statuses exist (e.g. Pending, Completed, Failed) and do any require special handling?

**IDEAS**

* **Lockout policy:** after 3 failed login attempts to prevent brute-force.
* **Automated emails/SMS:** notifications on withdrawal submission and completion.
* **Transaction History:** Could include filtering/sorting options for transactions

**Additional Considerations**

* **Edge Cases:** First day of month, last working day, leap year dates
* **UI/UX:** Responsiveness across devices, accessibility
* **Performance:** Loading times with many transactions
* **Security:** Test session timeout, unauthorized access attempts

1. **Feature Chain** This feature chain represents the core user journey, with each feature building upon the previous ones to create a complete experience for employees accessing their earned wages.

1. **Analyze features and draft Test Scenarios**
   1. **User Authentication Feature Test Scenarios**

| **Scenarios** | **Description** |
| --- | --- |
| **Scenario 1.1:** Successful Login with Valid Phone Number | The employee enters a valid phone number in the correct format. The system should verify the phone number against the database records and send a one-time password (OTP) to the registered phone number. After the employee enters the correct OTP, the system should authenticate the user and redirect them to the dashboard or home screen. This test verifies that legitimate users can access their accounts through the proper authentication flow. |
| **Scenario 1.2:** Login Attempt with Invalid Phone Number | The employee enters a phone number that is not registered in the system. The system should display an appropriate error message indicating that the phone number is not recognized and provide guidance on next steps (such as contacting HR or registering). This test ensures that unauthorized users cannot proceed with the authentication process. |
| **Scenario 1.3:** Login with Incorrect OTP | After entering a valid phone number and receiving an OTP, the employee enters an incorrect OTP. The system should display an error message and allow the user to retry. After a predetermined number of failed attempts (e.g., 3), the system should temporarily lock the account or require additional verification. This test verifies that the OTP mechanism provides adequate security. |
| **Scenario 1.4:** OTP Expiration | The employee enters a valid phone number and receives an OTP but waits until after the OTP expiration time (typically 5-10 minutes) before entering it. The system should reject the expired OTP and offer to resend a new one. This test ensures that temporary authentication codes cannot be used indefinitely. |
| **Scenario 1.5:** Phone Number Format Validation | The employee enters a phone number in various incorrect formats (too few digits, too many digits, including letters, etc.). The system should validate the input and provide clear feedback on the expected format. This test verifies that the system properly validates input before attempting authentication.  **QUESTION:** Is there a specific phone number format requirement (country code, number of digits) that should be enforced during validation?  **IDEA:** Implement a "Remember this device" option during login to simplify future authentication for employees using personal devices, reducing friction in the login process while maintaining security. |

* 1. **Dashboard/Home Feature Test Scenarios**

| **Scenarios** | **Description** |
| --- | --- |
| **Scenario 2.1:** Accurate Earned Wage Calculation on 12th Working Day | After successful login, the employee should see their earned wage calculated based on being on the 12th working day of the month. The system should display the correct amount based on the employee's salary, working days in the month, and days worked so far. This test verifies that the core calculation functionality works correctly under normal circumstances. |
| **Scenario 2.2:** Wage Display with Different Employment Types | Test the earned wage display for different types of employees (full-time, part-time, hourly, salaried) to ensure the calculation logic accommodates various employment arrangements. The system should apply the appropriate calculation method based on the employee's contract type. This test ensures that all employee types can use the feature correctly. |
| **Scenario 2.3:** Wage Display with Overtime or Bonuses | The employee has worked overtime or earned bonuses during the first 12 days of the month. The system should accurately include these additional earnings in the calculated earned wage. This test verifies that the system accounts for variable compensation components. |
| **Scenario 2.4:** Wage Display with Deductions or Advances | The employee has mandatory deductions (taxes, benefits) or has already taken an advance. The system should correctly calculate the net earned wage after accounting for these factors. This test ensures that the displayed amount reflects what is actually available for withdrawal. |
| **Scenario 2.5:** Wage Information Refresh/Update | Test how and when the earned wage information updates - for example, if the employee views it on day 12, then logs in again on day 13, the amount should be recalculated. This test verifies that the system maintains current information rather than caching outdated calculations.  **QUESTION:** How frequently is the earned wage calculation updated? Is it real-time, daily, or at another interval?  **IDEA:** Include a visual representation (such as a progress bar or pie chart) showing the proportion of monthly salary earned so far, helping employees better understand their financial position at a glance. |

* 1. **Withdraw Earned Wage Feature Test Scenarios**

| **Scenarios** | **Description** |
| --- | --- |
| Scenario 3.1: Successful Full Earned Wage Withdrawal | The employee initiates a withdrawal for the entire available earned wage amount. The system should process the request, confirm the transaction, and update the available balance to zero. This test verifies that the core withdrawal functionality works for maximum amounts. |
| Scenario 3.2: Successful Partial Earned Wage Withdrawal | The employee initiates a withdrawal for less than the total available earned wage. The system should process the request, confirm the transaction, and update the available balance to reflect the remaining amount. This test ensures that partial withdrawals are handled correctly. |
| Scenario 3.3: Withdrawal Exceeding Available Amount | The employee attempts to withdraw more than their available earned wage. The system should prevent the transaction and display an appropriate error message explaining the maximum available amount. This test verifies that the system enforces proper limits on withdrawals. |
| Scenario 3.4: Withdrawal with Processing Fee | If the system charges a fee for withdrawals, test that the fee is correctly calculated, displayed to the user before confirmation, and included in the transaction details. This test ensures transparency in the withdrawal process. |
| Scenario 3.4: Withdrawal with Processing Fee and Discount Voucher | ?? |
| Scenario 3.5: Withdrawal to Different Payment Methods | Test withdrawals to different payment destinations (bank account, debit card, mobile wallet, etc.) to ensure the system can process transactions through various channels. This test verifies flexibility in how employees can receive their funds. |
| Scenario 3.6: Withdrawal Cancellation | The employee initiates a withdrawal but cancels it before final confirmation. The system should abort the transaction and maintain the original available balance. This test ensures that users can change their minds before committing to a withdrawal.  **QUESTION:** Are there any daily, weekly, or monthly limits on the number or amount of withdrawals an employee can make?  **IDEA:** Implement a "quick withdrawal" feature that allows employees to set up predefined withdrawal amounts (e.g., 25%, 50%, 75% of available funds) to streamline the process for frequent users. |

* 1. **Transaction History Feature Test Scenarios**

| **Scenarios** | **Description** |
| --- | --- |
| Scenario 4.1: Transaction History Display | After completing one or more withdrawals, the employee accesses the transaction history. The system should display all transactions with relevant details (date, amount, status, payment method, etc.) in chronological order. This test verifies that the transaction record-keeping functions properly. |
| Scenario 4.2: Transaction Filtering and Sorting | The employee applies filters (date range, amount range, status) or sorting options to the transaction history. The system should update the display according to the selected criteria. This test ensures that users can effectively navigate their transaction history. |
| Scenario 4.3: Transaction Details View | The employee selects a specific transaction from the history. The system should display detailed information about that transaction, including any processing fees, payment method details, and processing timestamps. This test verifies that detailed transaction information is accessible. |
| Scenario 4.4: Empty Transaction History | A new employee who has never made a withdrawal views the transaction history. The system should display an appropriate message indicating that no transactions have been made yet. This test ensures that the system handles edge cases gracefully. |
| Scenario 4.5: Transaction Receipt Generation | The employee requests a receipt or statement for a specific transaction. The system should generate a downloadable or printable document containing all relevant transaction details. This test verifies that users can obtain formal records of their transactions. |
| Scenario 4.6: Failed Transaction Display | If a withdrawal was attempted but failed (due to technical issues, insufficient funds, etc.), the system should include this in the transaction history with appropriate status indicators and explanation. This test ensures that the transaction history is comprehensive.  **QUESTION:** How long are transaction records retained and visible to employees in the system?  **IDEA:** Include a monthly summary feature that aggregates withdrawal data and presents insights on withdrawal patterns, helping employees better understand their financial behaviors. |

* 1. **Cross-Feature Test Scenarios**

| **Scenarios** | **Description** |
| --- | --- |
| Scenario CF.1: End-to-End User Journey | Test the complete user journey from login through viewing earned wage, making a withdrawal, and checking the transaction history. All features should work together seamlessly, with data consistency maintained throughout the process. This test verifies the overall integration of the application components. |
| Scenario CF.2: Application Performance Under Load | Simulate multiple concurrent users performing various operations to test system performance and response times under load. The application should maintain acceptable performance levels even during peak usage periods. This test ensures that the system can handle real-world usage patterns. |
| Scenario CF.3: Data Consistency Across Features | After making a withdrawal, verify that the available earned wage amount and transaction history are updated consistently across all relevant screens. This test ensures that the system maintains data integrity throughout the user experience.  **IDEA:** Implement a notification system that alerts employees when new wages are earned or when pending withdrawals are completed, enhancing the user experience through proactive communication. |

1. **High Level Design Test Scenarios (HLTS) and Test Traceability Matrix (TCM)**

Please refer to the contents in the package. -> ..\Part 1\test specification\[**HLTS\_salary\_hero.xlsx**](Part%201/test%20specification/HLTS_salary_hero.xlsx)

1. **Low Level Design Test Cases (LLTC)**

Please refer to the contents in the package. -> ..\Part 1\test specification\[**LLTC\_salary\_hero.xlsx**](Part%201/test%20specification/LLTC_salary_hero.xlsx)

1. **Salary Hero Test Documentation Manual**

Please refer to the contents in the package. -> ..\Part 1\test specification\[**Salary Hero Test Documentation Manual.md**](Part%201/test%20specification/Salary%20Hero%20Test%20Documentation%20Manual.md)

**Part 2: Automated test assignment (Bonus)***To be completed a new link* (the original one is broken)