American International University-Bangladesh (AIUB)

Department of Computer Science

Faculty of Science & Technology (FST)

PROJECT TITLE

AIUB Canteen Management System

Semester: Fall 25-26

Group: ***5 (Turtlers)***		***Section:***
SL	**Student Name**	**Student ID**
1	Tahseen Habib Maliha	21-44402-1
2	Faysal Ahmed Shovo	23-50529-1
3	Sayem Mahmud	23-53439-3
4	Aswat Shahriar	23-55250-3
5		

Table of Contents

1. PROJECT PROPOSAL

1.1 Background to the Problem

The canteen at American International University–Bangladesh (AIUB)serves many students, faculty members, and university staff every day. Currently, most of the canteen operations are handled manually, which creates several problems, such as:
Long queues during peak hours

Slow order and payment processing

Miscommunication between customers and canteen staff

Difficulty in managing stock and maintaining accurate sales records

Lack of data-driven decision-making for improving services

These issues cause wasted time, poor service experience, and operational inefficiency. As the number of students and staff continues to grow, the traditional manual system is no longer sufficient to handle daily demand.
To solve these problems, the proposed AIUB Canteen Management System (ACMS)will introduce a digital platform that automates daily canteen operations. Users will be able to view menus, place orders, make digital payments, and track order status. At the same time, canteen staff and management will be able to manage menus, monitor inventory, and generate reports efficiently. This system aims to make the canteen faster, smarter, and more reliable for everyone.

Target Users
The ACMS is designed for the following users:
a. Students

- View daily and weekly food menus
- Place online food orders
- Make digital payments
- Receive order-ready notifications
- Provide feedback and ratings

b. Faculty Members

- Pre-order meals to avoid rush hours
- Place flexible food orders
- Track order history and payments

c. University Staff

- Place online orders quickly
- Choose dine-in or takeaway options
- Manage their personal profiles and order records

d. Canteen Employees

- View real-time incoming orders
- Update order preparation and serving status
- Manage food availability

e. Canteen Manager

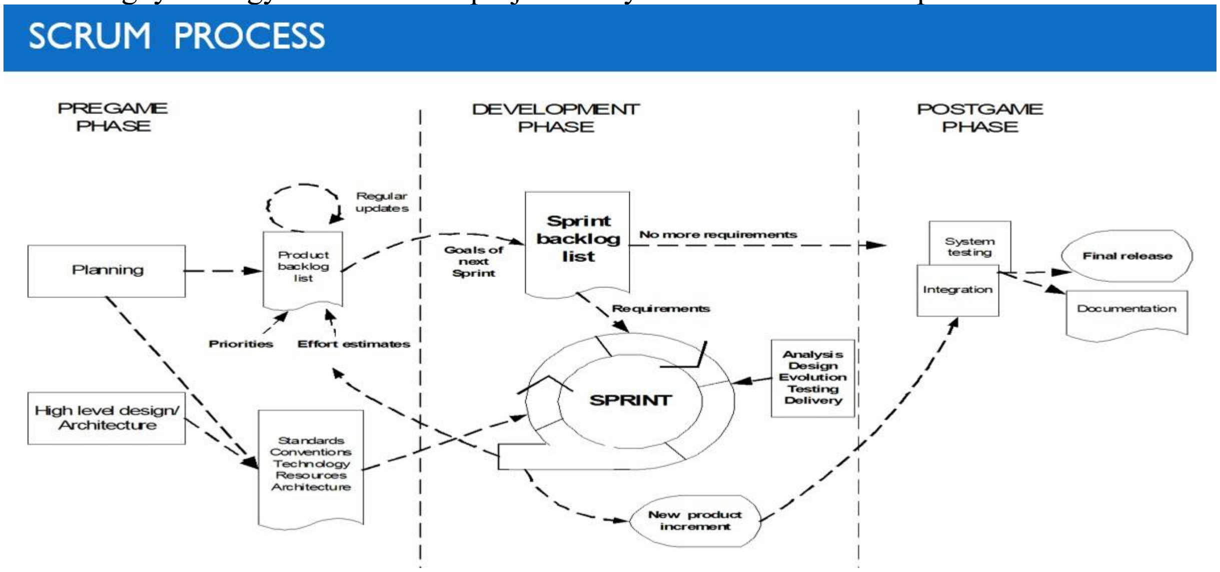
- Add, update, or remove food items from the system
- Manage food categories and pricing
- Monitor employee schedules and performance
- View sales and stock reports

f. Admin

- Manage system users and roles
- Control system access and security
- Monitor system activities and logs
- Generate system-wide reports
- Perform data backup and recovery

1.2 Selection of Process Model

Scrum is an adaptive, quick, and self-organizing product development process that originated from a rugby strategy. It divides the project lifecycle into three distinct phases:



It divides the project lifecycle into three distinct phases:

1. **Pre-game Phase**

- Planning: A Product Backlog list is created containing all known requirements, which are prioritized and estimated.
- Architecture: High-level design and system architecture are planned based on the backlog.

2. **Game (Development) Phase**

   - This phase is treated as a "black box" where development occurs in iterative cycles called Sprints.
   - Each Sprint lasts one to four weeks and includes traditional development steps like analysis, design, and delivery to produce a new product increment.

3. **Post-game Phase**

   - This phase begins when requirements are completed. It involves final integration, system testing, and documentation to prepare the system for final release.

Key Practices & Roles

- **Backlogs**: The Product Backlog lists all requirements, while the Sprint Backlog lists specific items selected for implementation in the current Sprint.
- **Meetings:** The process relies on Sprint Planning Meetings to set goals, Daily Scrum meetings to track progress, and Sprint Review meetings to present the working product.
- **Roles:** Key roles include the Scrum Master (ensure rules are followed), the Product Owner (manages the backlog), and the Scrum Team (self-organizing development team).

Why we select Scrum process model:

We have selected the Agile Scrum model for the ACMS project because it directly addresses the operational challenges of the canteen environment. The justification for this selection is based on the following factors:

1. **Adaptability to Changing Requirements**: The requirements for a canteen system can evolve. For instance, feedback from the canteen manager might reveal a need for a new inventory alert feature that was not originally planned. Agile allows us to incorporate these changes in the next Sprint without restarting the entire development process.
2. **Early Delivery of Solutions (MVP):** The background problem highlights urgent issues like "Long queues" and "Slow processing." Agile enables us to release a Minimum Viable Product (MVP)—such as a basic ordering system—early in the timeline. This allows the canteen to start solving the queue problem immediately while we continue to develop advanced features like reporting and stock management.
3. **Continuous Feedback and Testing**: Since the system serves a large user base of students and faculty, user experience is critical. Agile facilitates continuous feedback loops. After every Sprint, we can demonstrate the latest version to potential users to ensure the interface is intuitive and meets their expectations, reducing the risk of

"miscommunication" mentioned in the problem statement.

4. **Risk Mitigation:** Transitioning from a manual to an automated system carries technical risks. By developing in small increments, errors and bugs are identified and resolved within the Sprint they occur, rather than accumulating until the end of the project. This ensures a more stable and reliable final product.

2. SOFTWARE REQUIREMENTS SPECIFICATIONS (SRS) / PRODUCT REQUIREMENTS DOCUMENT (PRD)

2.1 Scopes and Features

Users:

1. Student
2. Faculty
3. Staff
4. Canteen Employee
5. Canteen Manager
6. Admin

Scopes:

1) Profile Management

Features:

a) All users can create new profiles with valid information.

b) All users can login using a valid username and password.

c) After successful login, all users can view their dashboards.

d) All users can view their own profile details.

e) All users can update their personal information with valid data.

f) All users can change their password.

g) All users can recover password after verification.

2) User Management

Features:

a) Admin can add, update, or remove system users.

b) Admin can assign roles and permissions to users.

c) Admin can activate or deactivate user accounts.

d) Admin can view the full user list.

3) Menu & Food Management

Features:

a) Canteen employees and manager can add new food items to the system.

b) Students, faculty, and staff can view the full food menu.

c) Students, faculty, and staff can search food items by name, category, or price.

d) Canteen employees and manager can update food information.

e) Manager can delete food items from the menu.

f) Canteen employees can update food availability status.

4) Order Management

Features:

a) Students, faculty, and staff can add food items to the cart.

b) Students, faculty, and staff can place an order.

Status:

Pending Order

c) After successful payment, the status changes automatically.

Status:

Paid Pending Order

d) Canteen employees can view all paid pending orders.

e) Canteen employees prepare food and update order status.

Status:

Preparing Order

f) When the order is ready, employees notify users.

Status:

Ready for Pickup

g) After food is collected, the system updates order status.

Status:

Completed Order

5) Payment Management

Features:

a) Students, faculty, and staff can pay using cash, card, or university wallet.

b) The system generates an e-receipt after successful payment.

c) Manager can view all transaction records.

d) Admin can audit payments and revenue reports.

6) Employee Management

Features:

a) Manager and admin can add, update, or remove canteen employees.

b) Manager can assign work roles to employees.

c) Employees can update their working schedule and attendance.

Untitled 6

d) Admin can view employee activity summaries.

7) Feedback & Rating Management

Features:

a) Students, faculty, and staff can submit feedback and ratings.

b) Manager can view and respond to feedback.

c) Admin can analyze feedback trends.

8) Report & Sales Management

Features:

a) Manager can view daily, weekly, and monthly sales reports.

b) Admin can view total revenue and profit statistics.

c) Manager can export reports for decision-making.

2.2 User Story Table

A/An	I want to	So that	Acceptance Criteria
User (Student / Faculty / Staff)	**Register**	I can enroll myself in the system and access canteen services digitally.	User must input Name, ID, Email, and Password. An OTP is sent to the user's mobile/email for verification.
	Login	I can securely access my private dashboard and order history.	The system authenticates me. After successful login, the data associated with the user is accessible.
	View Dashboard	I can easily navigate to different features like menu or profile after logging in.	Show side menu with navigation buttons and key status (e.g., wallet balance).

	Manage Profile	I can keep my personal details current and secure, and regain access if credentials are lost.	*Includes:* View Profile, Update Profile, Change Password. *Extends:* Allows OTP verification for password recovery.
	View Menu	I can decide what to eat based on current offerings.	Show categorized list of food items with prices. Items marked "Out of Stock" are visually distinct or hidden.
	Search Food	I can find specific cravings without scrolling through the whole list.	Show search bar taking keywords. Show filtered results matching the query.
	Add to Cart	I can build a meal order with multiple items.	Show "Added to cart" popup message. Update cart icon counter.
	Place Order	The kitchen knows exactly what to prepare for me, and the transaction is recorded.	Show order summary. Status changes to "Pending" upon confirmation.
	Make Payment	The transaction is completed and the order is confirmed.	Redirect to payment gateway. Order status changes to "Paid Pending".
	View E-Receipt	I have proof of purchase in case of disputes.	Generate digital receipt with Transaction ID, Date, and Items after successful payment.
	Give Feedback	The management can improve food quality based on my experience.	Show form taking 1–5-star rating and text comment. Submitting form gives success notification.
Canteen Employee	**View Incoming**	I can prioritize cooking based on order arrival	Show real-time list of "Paid Pending" orders. Show count of total pending orders.

	Orders	time.	
	Update Order Status	The customer is notified when their food is cooking or ready.	Change status (e.g., 'Pending' to 'Ready'). System triggers notification to the specific user.
	Manage Menu Items	I can fix minor errors in price or description quickly.	Show edit form for specific food items. User inputs new price/name and saves.
	Update Availability	Customers do not order items that have run out of stock.	Toggle button for "In Stock" / "Out of Stock" on food items. Menu updates immediately for all users.
	Update Schedule	My shift duties and availability are correctly recorded by management.	User can input and update their working schedule and attendance status.
Canteen Manager	**Manage Menu Items**	I can introduce new dishes or remove old ones.	User must input Name, Price, Category, and Image for new items. Success notification upon submission.
	Manage Food Categories	I can organize the menu into logical sections (e.g., Breakfast, Lunch).	Show list of categories with options to Create or Delete.
	Monitor Orders	I can identify bottlenecks in the kitchen workflow.	Show master list of all orders (Pending, Cooking, Completed, Cancelled) in real time.
	View Transactions	I can reconcile cash flow with bank statements.	Show detailed log of every payment (Time, ID, Amount, Method).

	Manage Employees	I can ensure only authorized staff have access to the system.	Show employee list table with "Add", "Edit", and "Delete" buttons, and options to Assign Roles.
	View Feedback	I can address complaints or reward good performance.	Show list table of user ratings and comments, with options to Respond To Feedback.
	View Sales Reports	I can analyze financial performance and popular items, and keep offline records.	Show daily/monthly sales tables and total profit calculation. Allows downloading the report as PDF or Excel file.
Admin	**Login**	I can access the administrative backend for oversight and management.	The system authenticates me with administrator credentials.
	Manage Users	I can ensure only authorized university and canteen staff have access to the system.	Includes: View Users, Add User, Update User, Delete User. Shows a list of all
			system users (including Canteen Manager).
	Assign Roles	I can set correct access levels for new or existing staff.	Interface allows selecting a user and assigning a specific role (e.g., Manager, Employee).
	Activate / Deactivate Accounts	I can temporarily restrict or restore system access for any user.	Toggle button/option to change account status from "Active" to "Inactive."

	View System Logs	I can monitor user and system actions for security and troubleshooting.	Show time-stamped log entries of system events.
	Monitor Activities	I can proactively audit key system operations and user behavior.	Filterable interface to track specific user or transaction activities within the logs.
	Generate Reports	I can obtain high-level financial and operational summaries for the university.	Show aggregated data and statistics (e.g., total revenue, employee performance summaries).
	Backup Data	I can secure system information and quickly recover it after any failure.	Initiates the process to create a secure, recent copy of the entire system database. Restore Data uses the latest backup file.

2.3 Requirements Traceability Matrix

Requirement ID	**Test Case ID**	**Test Case**	**Expected Test Result**	**Design Status**	**Execution Status**	**Test Validation Status**
REQ 1	T1	Verify Login for admin with	Redirect to Admin	POSITIVE	PASS	PASS

Requirement ID	**Test Case ID**	**Test Case**	**Expected Test Result**	**Design Status**	**Execution Status**	**Test Validation Status**
		valid credentials	Dashboard			
REQ 2	T2	Verify Admin can add a new user	New user successfully added to the system	POSITIVE	PASS	PASS
REQ 3	T3	Verify Admin can assign user roles	User role updated successfully	POSITIVE	PASS	PASS
REQ 4	T4	Verify Admin can create canteen employee account	Employee account created successfully	POSITIVE	PASS	PASS
REQ 5	T5	Verify Admin can assign roles to employees and managers	Roles successfully assigned to staff	POSITIVE	PASS	PASS
REQ 6	T6	Verify Registration for User (Student/Staff)	User account created successfully	POSITIVE	PASS	PASS
REQ 7	T7	Verify Secure login for user with valid credentials	Redirect to User Dashboard	POSITIVE	PASS	PASS

Requirement ID	Test Case ID	Test Case	Expected Test Result	Design Status	Execution Status	Test Validation Status
REQ 8	T8	Verify Login for Canteen Manager	Redirect to Manager Dashboard	POSITIVE	PASS	PASS

2.3.1 Functional Requirements

4.1 General User Requirements (Student, Faculty, Staff)

ID	Requirement
1.0	**User Home Page - All features availability**
1.1	**Registration**
1.1.1	Taking name
1.1.2	Taking email address
1.1.3	Taking User ID
1.1.4	Taking password
1.1.5	Taking confirm password
1.1.6	Taking phone number
1.1.7	Submitting the form
1.1.8	Resetting the form
1.2	**Login**
1.2.1	Taking User ID
1.2.2	Taking password

1.2.3	Recovery Password Option
1.2.3.1	Submitting email address
1.2.3.2	Getting OTP via email
1.2.3.3	Submission of OTP
1.2.3.4	Taking new password
1.2.3.5	Taking confirm new password
1.2.4	Submission of the login

1.3	**Dashboard**
1.3.1	Side Menu
1.3.1.1	Dashboard button
1.3.1.2	Manage Profile Button
1.3.1.2.1	Show name
1.3.1.2.2	Show email
1.3.1.2.3	Show User ID
1.3.1.2.4	Update Personal Information Button
1.3.1.2.4.1	Show existing phone number and taking new phone number
1.3.1.2.4.2	Submission of the updated info
1.3.1.2.4.3	View profile
1.3.1.2.5	Change Password Button
1.3.1.2.5.1	Taking current password
1.3.1.2.5.2	Taking new password
1.3.1.2.5.3	Taking retype new password
1.3.1.2.5.4	Submitting the form
1.3.1.3	**View Menu Button**
1.3.1.3.1	Show Food Categories (Tabs)
1.3.1.3.2	Show Food Items List
1.3.1.3.3	**Search Food Item Button**
1.3.1.3.3.1	Taking food name keyword
1.3.1.3.3.2	Submitting the search

1.3.1.3.3.3	Show searched food items in a table
1.3.1.4	**Place Order Flow**
1.3.1.4.1	**Add to card (cart) button**
1.3.1.4.1.1	Show "Item Added" toast message
1.3.1.4.2	**View Cart Button**
1.3.1.4.2.1	Show selected items list table
1.3.1.4.2.2	**Apply Discount Button**
1.3.1.4.2.2.1	Taking coupon code
1.3.1.4.2.2.2	Validate and update total price
1.3.1.4.2.3	**Make Payment Button**
1.3.1.4.2.3.1	Select Payment Gateway
1.3.1.4.2.3.2	Confirm Payment
1.3.1.4.2.4	**View E-Receipt**
1.3.1.4.2.4.1	Show Transaction ID and Order Summary
1.3.1.5	**Give Feedback Button**
1.3.1.5.1	Taking star rating (1-5)
1.3.1.5.2	Taking feedback text
1.3.1.5.3	Submitting the form

4.2 Canteen Employee Requirements

ID	**Requirement**
2.0	**Dashboard**
2.1	**View Incoming Orders Button**
2.1.1.1	Show "Paid Pending" order list table
2.1.1.2	Show total pending count
2.1.1.3	**Update Order Status Button**
2.1.1.3.1	Select status (Cooking / Ready)
2.1.1.3.2	Submitting status update
2.1.1.3.3	Notify User (System Trigger)

2.1.2	**Manage Menu (Availability)**
2.1.2.1	Show all menu items list
2.1.2.2	**Update Availability Button**
2.1.2.2.1	Toggle In-Stock / Out-of-Stock
2.1.2.2.2	Save changes

4.3 Canteen Manager Requirements

ID	**Requirement**
3.0	**Dashboard**
3.1	**Manage Employee Button**
3.1.1.1	**Show available staff list table**
3.1.1.2	**Add Employee Button**
3.1.1.2.1	**Taking employee name, ID, and Role**
3.1.1.2.2	**Submitting the form**
3.1.1.3	**Delete Employee Button**
3.1.1.3.1	**Confirmation popup**
3.1.1.3.2	**Remove employee from database**
3.1.2	**Manage Menu Items Button**
3.1.2.1	**Add Food Item**
3.1.2.1.1	**Taking food name**
3.1.2.1.2	**Taking price**
3.1.2.1.3	**Taking category**
3.1.2.1.4	**Uploading image**
3.1.2.1.5	**Submitting the form**
3.1.2.2	**Manage Food Categories**
3.1.2.2.1	**Add/Delete Category names**
3.1.3	**Monitor Order Button**
3.1.3.1	**Show master order list (All statuses)**
3.1.4	**Report Section**
3.1.4.1	**View Sales Reports Button**

3.1.4.1.1	**Show daily/monthly sales table**
3.1.4.1.2	**Export Report Button**
3.1.4.1.2.1	**Generate PDF/Excel download**
3.1.4.2	**View Transaction Records Button**
3.1.4.2.1	**Show detailed payment log table**
3.1.4.3	**Review Feedback Button**
3.1.4.3.1	**Show user feedback table**

4.4 Admin Requirements

ID	**Requirement**
4.0	**Dashboard**
4.1	**Login**
4.1.1	Taking Admin credentials
4.1.2	Submission of the login
4.2	**Manage Users**
4.2.1	Show **View Users** list
4.2.2	**Add User**
4.2.3	**Update User**
4.2.4	**Delete User**
4.3	**Assign Roles** $\ll$include$\gg$ Manage Users
4.3.1	Select user and assign role
4.4	**Activate / Deactivate Accounts** $\ll$include$\gg$ Manage Users
4.4.1	Toggle account status
4.5	**View System Logs**
4.5.1	Show detailed log table
4.6	**Monitor Activities** $\ll$include$\gg$ View System Logs
4.6.1	Filter and search logs
4.7	**Generate Reports**

4.7.1	Generate requested report type
4.8	**Backup Data** $\ll$extend$\gg$ Restore Data
4.8.1	Initiate system backup
4.8.2	**Restore Data** (Extend)

2.3.2 Non-Functional Requirements

The non-functional requirements of the AIUB Canteen Management System (ACMS) define the quality attributes, performance standards, and operational constraints of the system. These requirements ensure that the system is reliable, efficient, and user-friendly for all stakeholders.

1. **Availability**

The system shall be available 24/7, except during scheduled maintenance periods. System uptime must be at least 99.5%, ensuring uninterrupted access for students, faculty, staff, and canteen employees during peak hours. Automated backup and recovery mechanisms shall be implemented to minimize downtime in case of system failure.

2. Performance

The system shall respond to user actions within 2 seconds under normal operating conditions. It must support at least 1,000 concurrent users without performance degradation. Order processing, payment confirmation, and notification delivery must occur in real time to ensure smooth canteen operations.

3. **Maintainability**

The system shall be designed using modular **architecture**, allowing easy updates, bug fixes, and feature enhancements. Proper documentation and coding standards shall be followed so that future developers can maintain and extend the system efficiently with minimal effort.

4. **User-Friendliness**

The system should provide a **simple, intuitive, and responsive user interface** accessible to users with minimal technical knowledge. Clear navigation, readable fonts, meaningful icons, and informative messages shall be used to enhance user experience across desktop and mobile devices.

5. **Efficiency**

The system should optimize resource usage, including server load and database operations. Automated processes will reduce manual workload, minimize errors, and improve order handling, inventory tracking, and report generation efficiency.

3. SOFTWARE DESIGN

3.1 System Design:

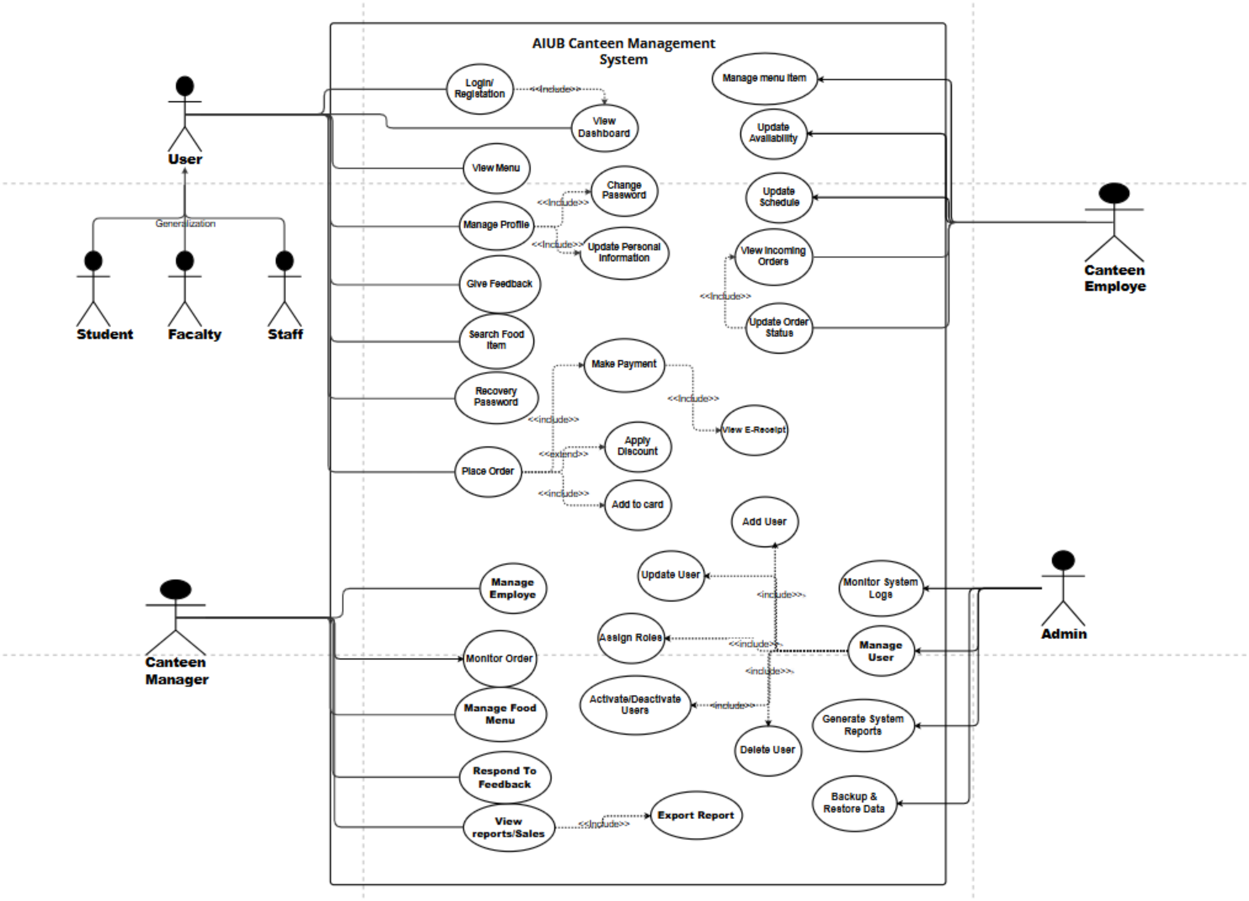


Figure1: Use Case Diagram

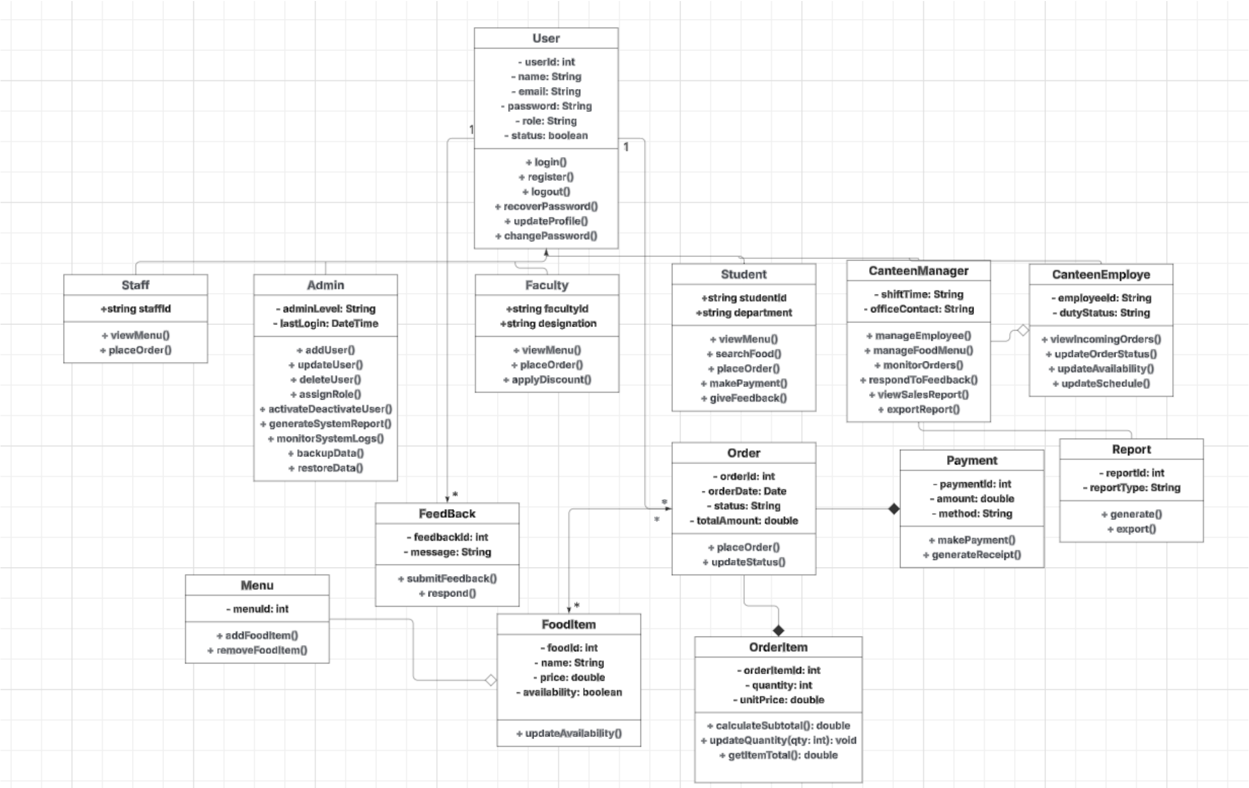


Figure2: Class Diagram

3.2 UI / Wireframe Design using Figma

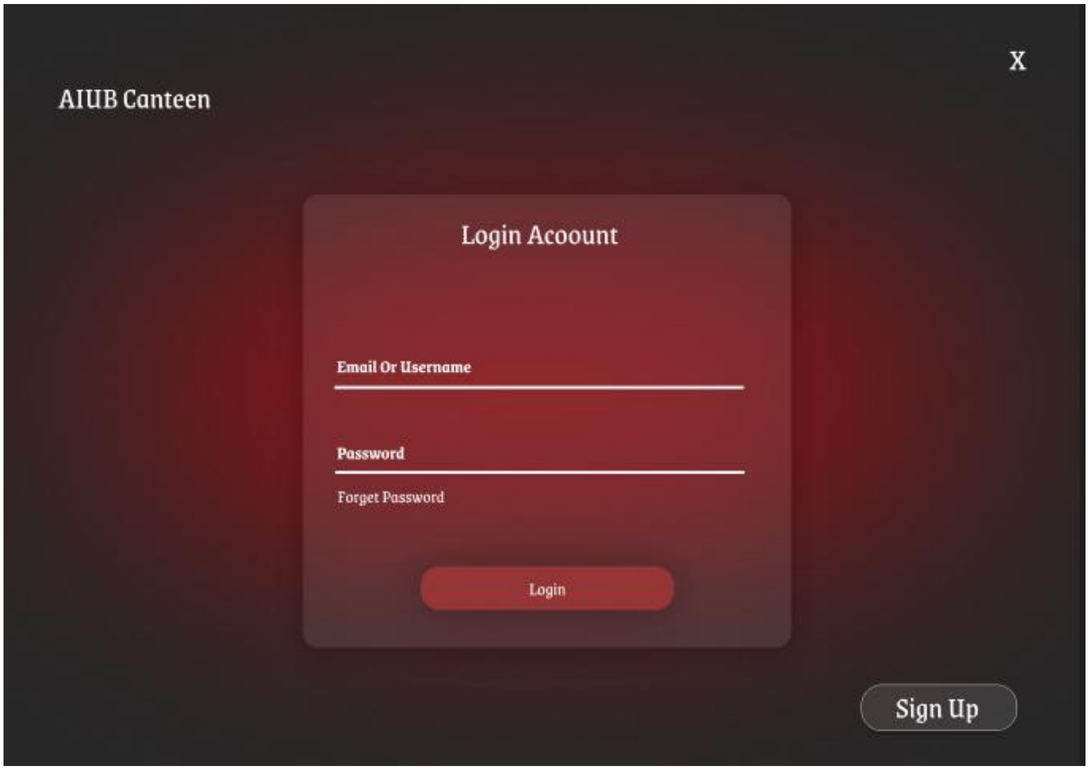


Figure3: Login page for user

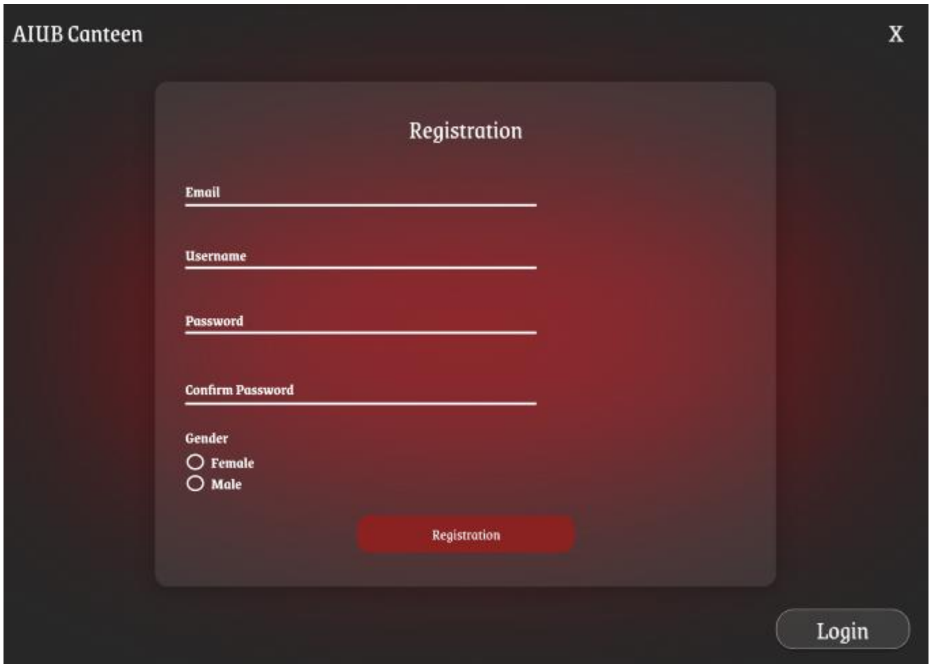
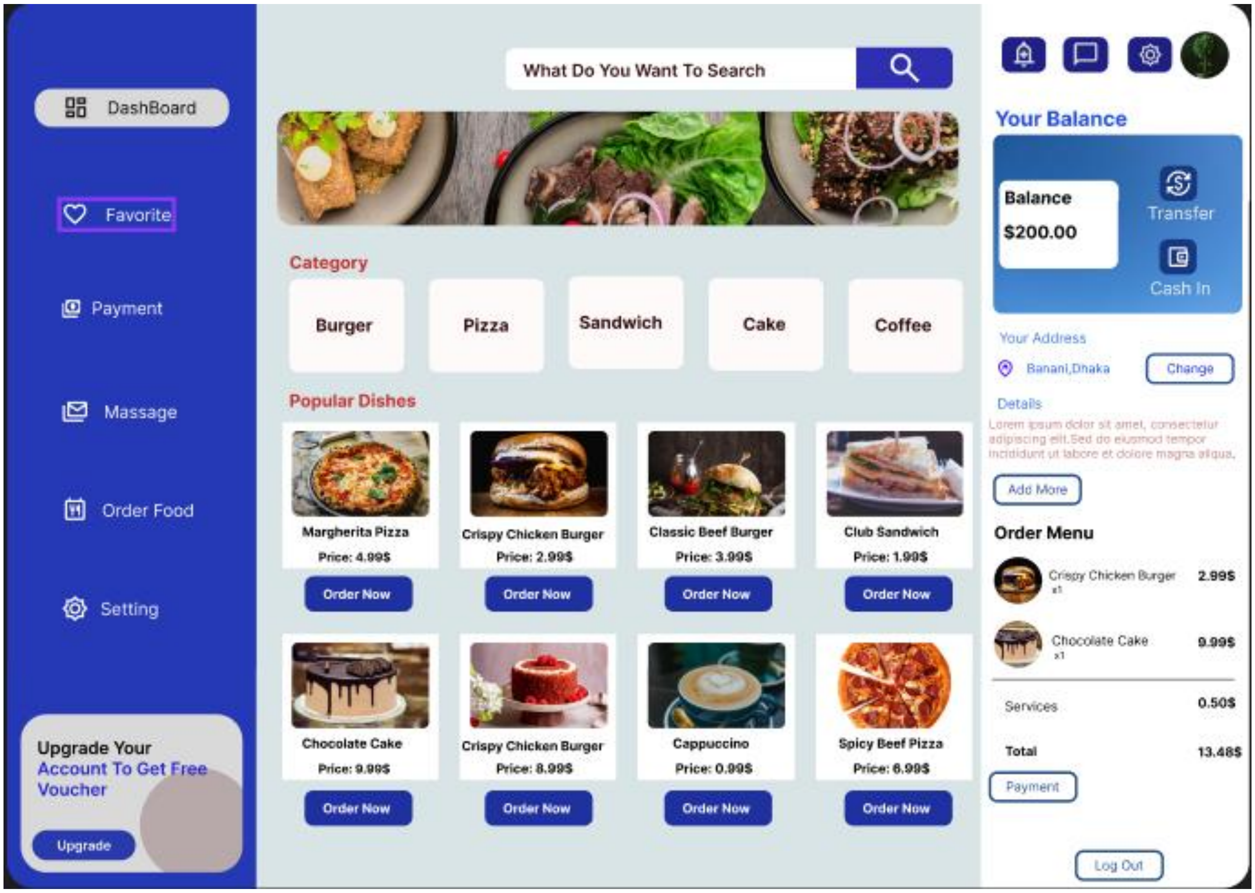


Figure4: Registration page for User



Margherita Pizza
Price: 4.99$

Crispy Chicken Burger
Price: 2.99$

Classic Beef Burger
Price: 3.99$

Club Sandwich
Price: 1.99$

Chocolate Cake
Price: 9.99$

Crispy Chicken Burger
Price: 8.99$

Cappuccino
Price: 0.99$

Spicy Beef Pizza
Price: 6.99$

Balance
$200.00

Your Address
Banani,Dhaka

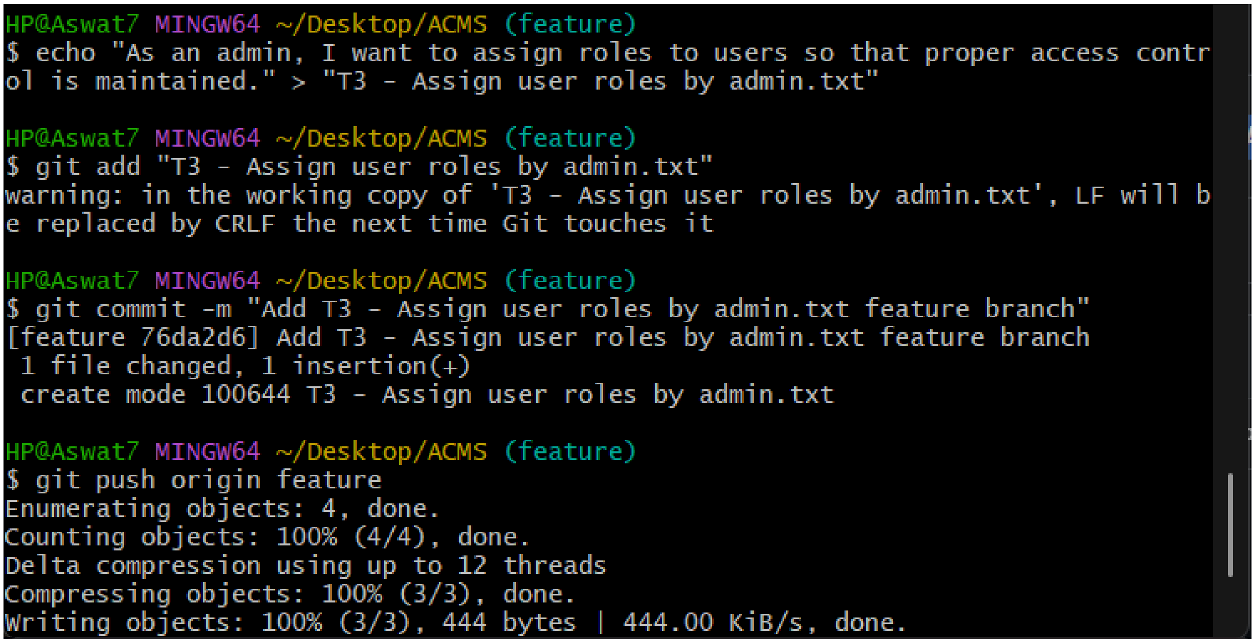
Details
Lorem ipsum dolor sit amet, consectetur adipiscing elit.Sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Order Menu

Crispy Chicken Burger x1	2.99$
Chocolate Cake x1	9.99$
Services	0.50$
Total	13.48$

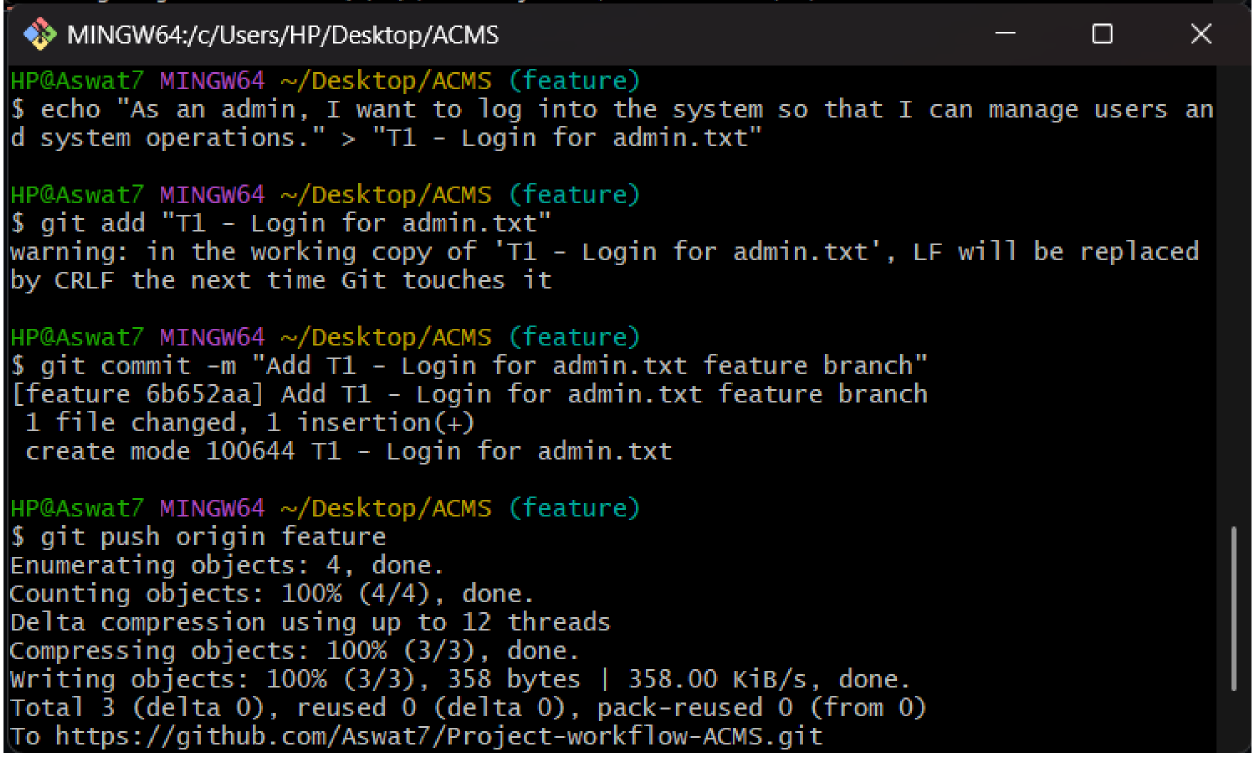
Figure5: User Dashboard

4. GIT WORKFLOW

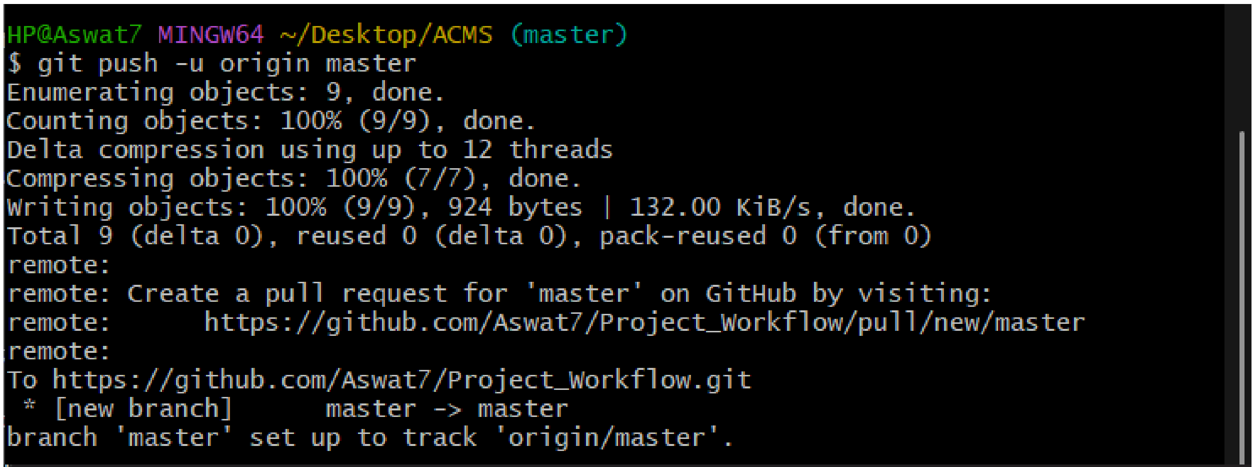


```
HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ echo "As an admin, I want to assign roles to users so that proper access contr
ol is maintained." > "T3 - Assign user roles by admin.txt"

HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ git add "T3 - Assign user roles by admin.txt"
warning: in the working copy of 'T3 - Assign user roles by admin.txt', LF will b
e replaced by CRLF the next time Git touches it

HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ git commit -m "Add T3 - Assign user roles by admin.txt feature branch"
[feature 76da2d6] Add T3 - Assign user roles by admin.txt feature branch
 1 file changed, 1 insertion(+)
 create mode 100644 T3 - Assign user roles by admin.txt

HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ git push origin feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 444 bytes | 444.00 KiB/s, done.
```

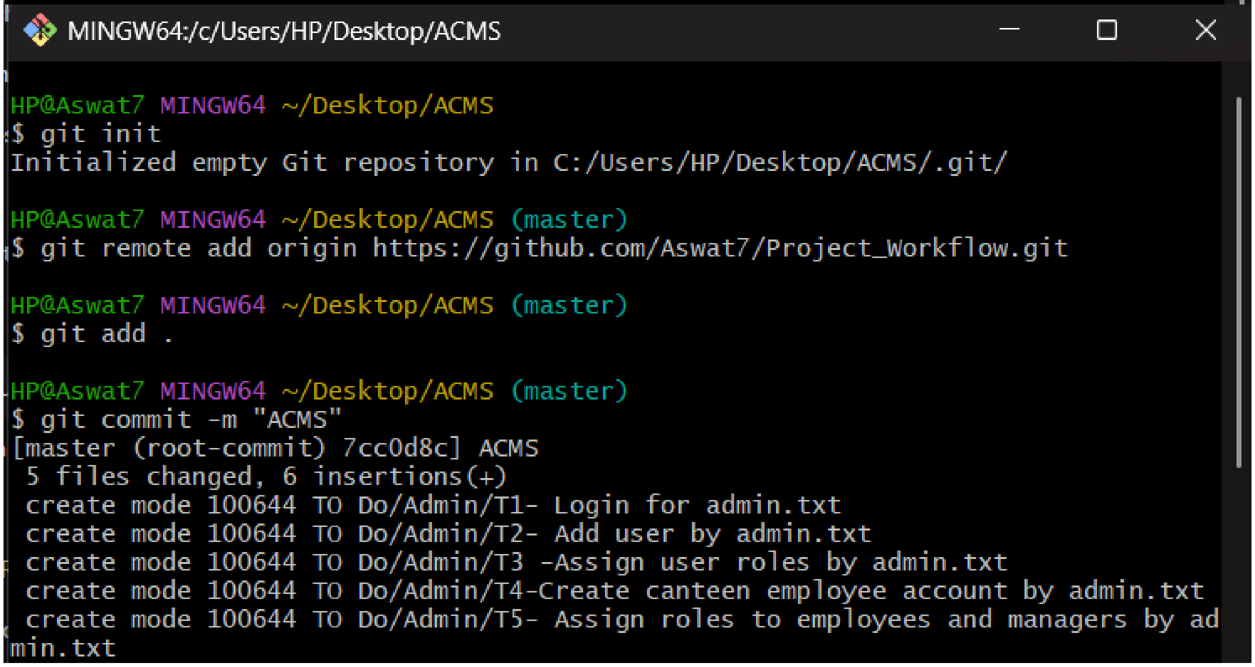
○



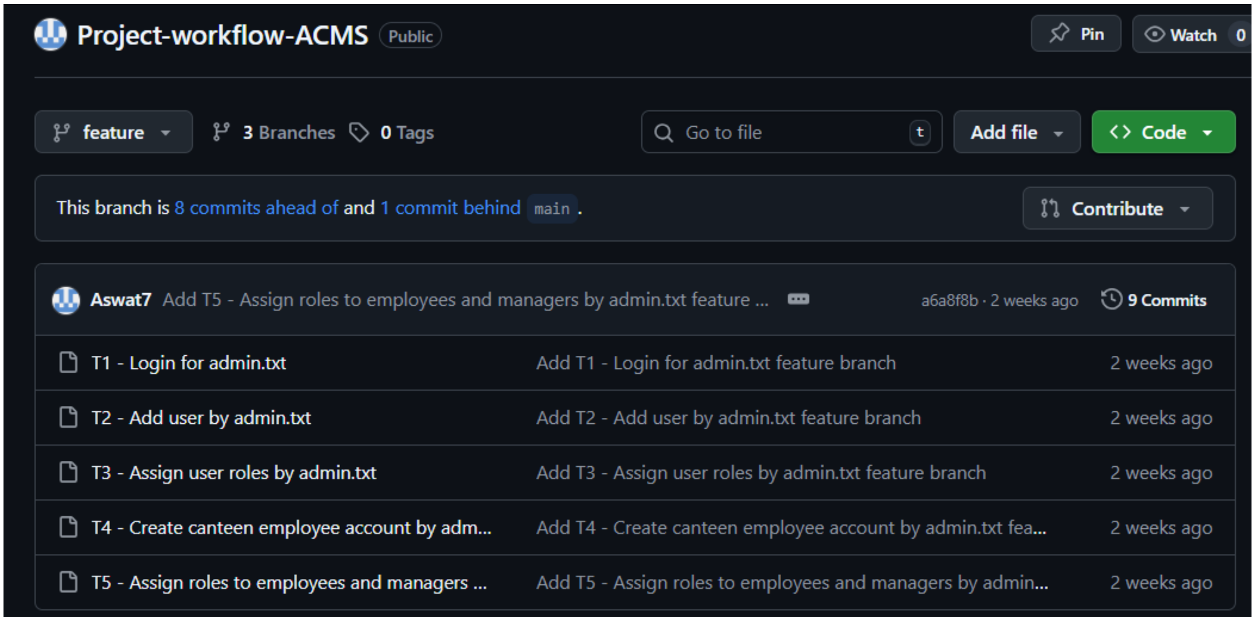
```
HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ echo "As an admin, I want to log into the system so that I can manage users an
d system operations." > "T1 - Login for admin.txt"

HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ git add "T1 - Login for admin.txt"
warning: in the working copy of 'T1 - Login for admin.txt', LF will be replaced
by CRLF the next time Git touches it

HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ git commit -m "Add T1 - Login for admin.txt feature branch"
[feature 6b652aa] Add T1 - Login for admin.txt feature branch
 1 file changed, 1 insertion(+)
 create mode 100644 T1 - Login for admin.txt

HP@Aswat7 MINGW64 ~/Desktop/ACMS (feature)
$ git push origin feature
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 358 bytes | 358.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Aswat7/Project-workflow-ACMS.git
```



```
HP@Aswat7 MINGW64 ~/Desktop/ACMS (master)
$ git push -u origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 924 bytes | 132.00 KiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/Aswat7/Project_Workflow/pull/new/master
remote:
To https://github.com/Aswat7/Project_Workflow.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```



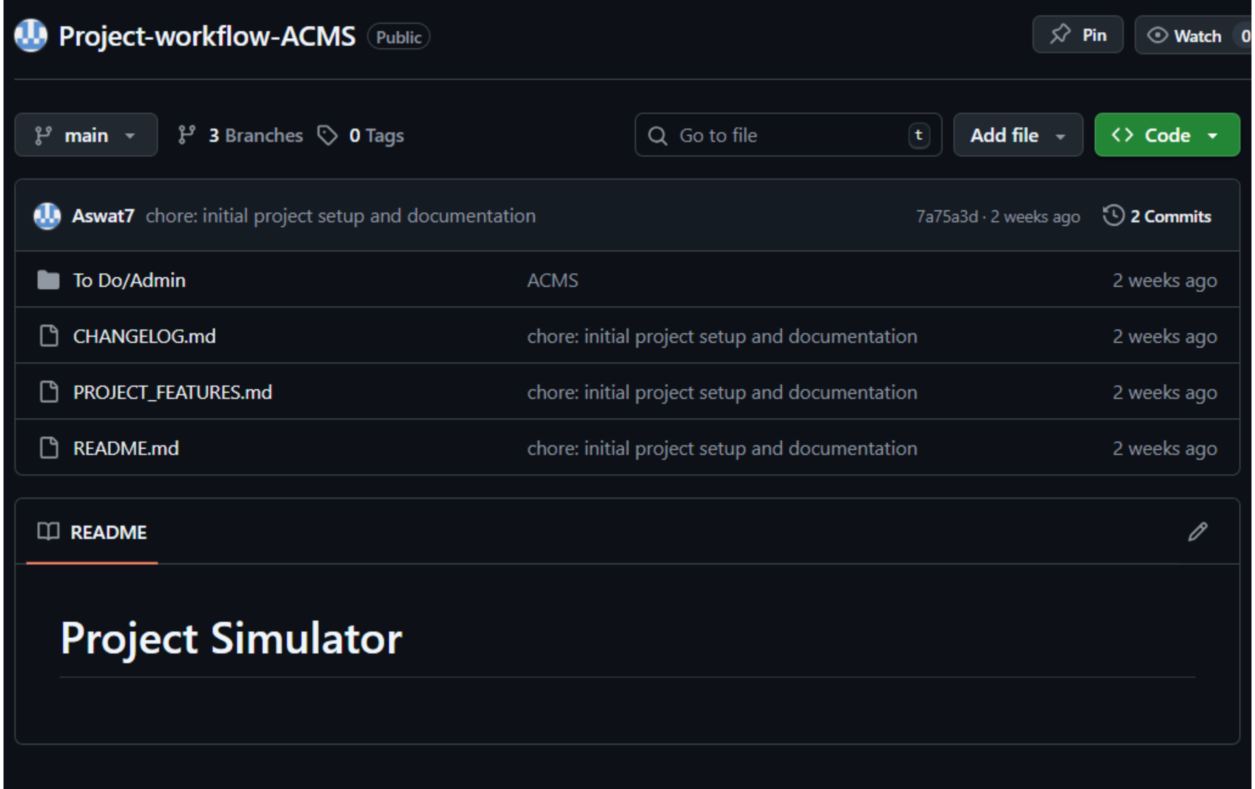
```
HP@Aswat7 MINGW64 ~/Desktop/ACMS
$ git init
Initialized empty Git repository in C:/Users/HP/Desktop/ACMS/.git/

HP@Aswat7 MINGW64 ~/Desktop/ACMS (master)
$ git remote add origin https://github.com/Aswat7/Project_Workflow.git

HP@Aswat7 MINGW64 ~/Desktop/ACMS (master)
$ git add .

HP@Aswat7 MINGW64 ~/Desktop/ACMS (master)
$ git commit -m "ACMS"
[master (root-commit) 7cc0d8c] ACMS
 5 files changed, 6 insertions(+)
 create mode 100644 TO Do/Admin/T1- Login for admin.txt
 create mode 100644 TO Do/Admin/T2- Add user by admin.txt
 create mode 100644 TO Do/Admin/T3 -Assign user roles by admin.txt
 create mode 100644 TO Do/Admin/T4-Create canteen employee account by admin.txt
 create mode 100644 TO Do/Admin/T5- Assign roles to employees and managers by ad
min.txt
```



This branch is 8 commits ahead of and 1 commit behind `main` .

Aswat7 Add T5 - Assign roles to employees and managers by admin.txt feature ... a6a8f8b · 2 weeks ago 9 Commits

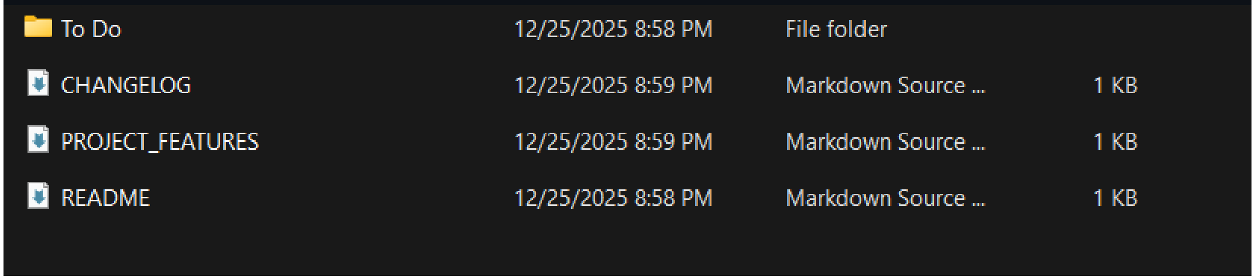
T1 - Login for admin.txt	Add T1 - Login for admin.txt feature branch	2 weeks ago
T2 - Add user by admin.txt	Add T2 - Add user by admin.txt feature branch	2 weeks ago
T3 - Assign user roles by admin.txt	Add T3 - Assign user roles by admin.txt feature branch	2 weeks ago
T4 - Create canteen employee account by adm...	Add T4 - Create canteen employee account by admin.txt fea...	2 weeks ago
T5 - Assign roles to employees and managers ...	Add T5 - Assign roles to employees and managers by admin...	2 weeks ago



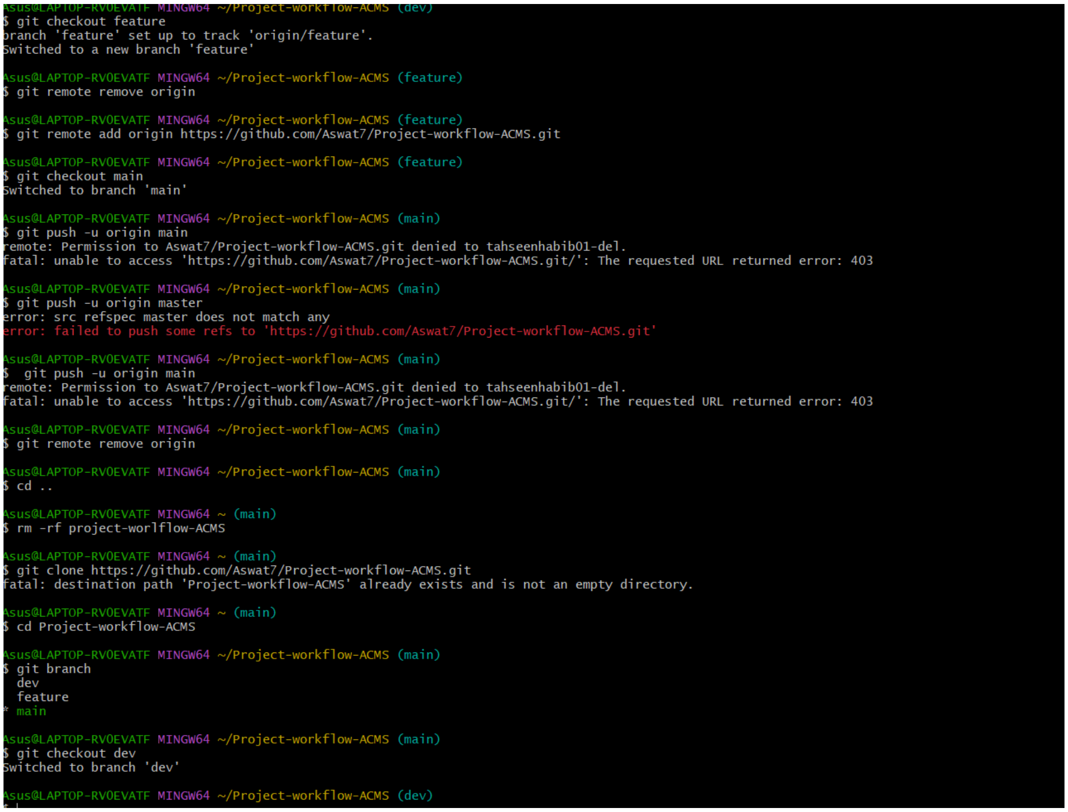
Aswat7 chore: initial project setup and documentation 7a75a3d · 2 weeks ago 2 Commits

To Do/Admin	ACMS	2 weeks ago
CHANGELOG.md	chore: initial project setup and documentation	2 weeks ago
PROJECT_FEATURES.md	chore: initial project setup and documentation	2 weeks ago
README.md	chore: initial project setup and documentation	2 weeks ago

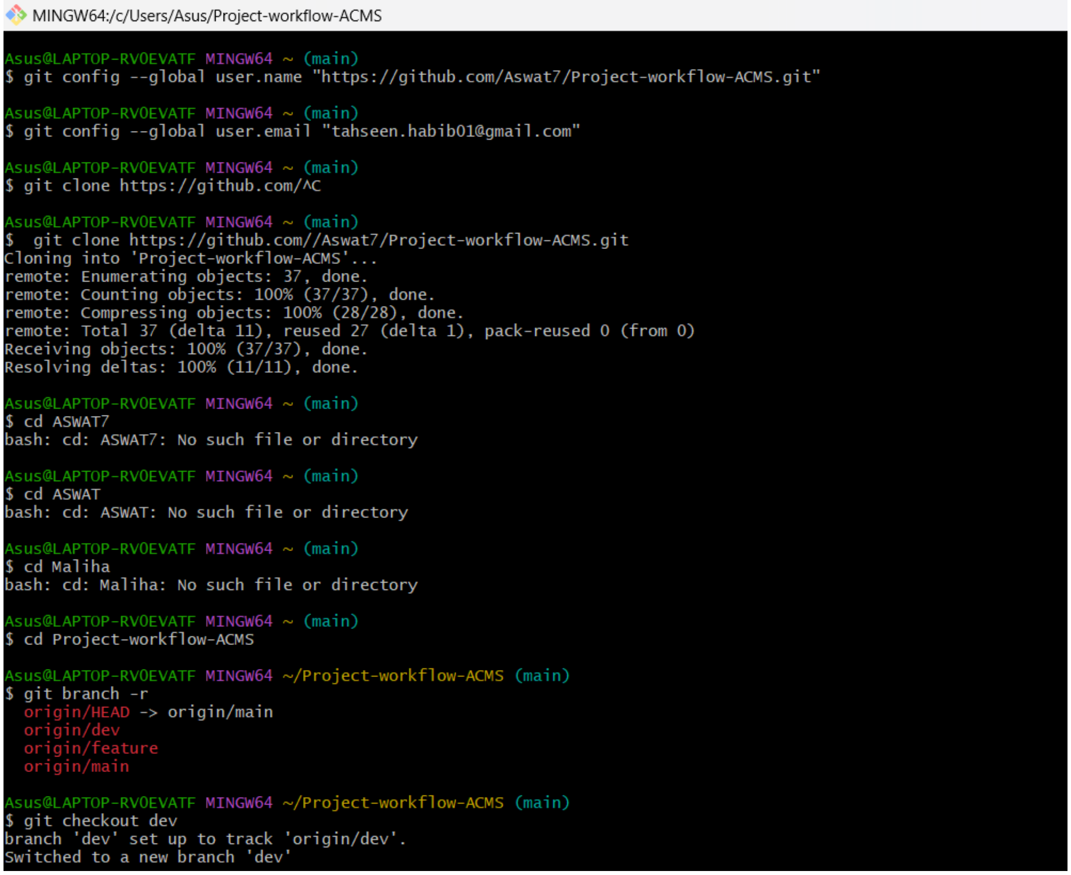
Project Simulator



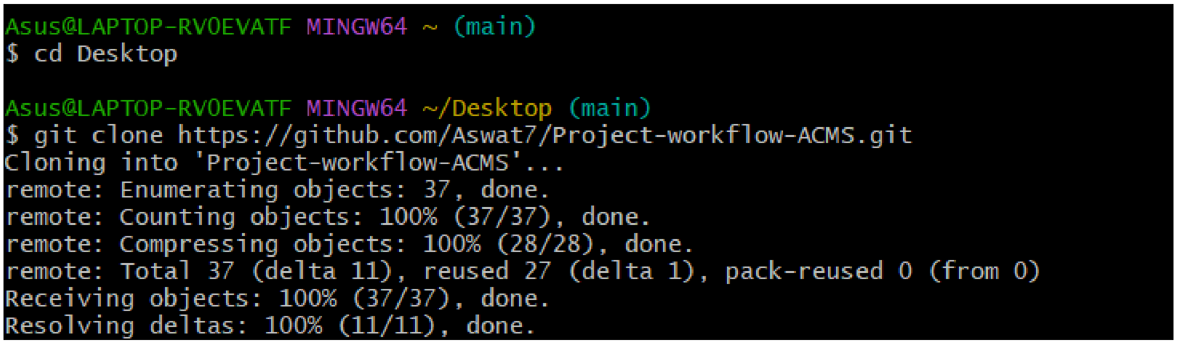
To Do	12/25/2025 8:58 PM	File folder	
CHANGELOG	12/25/2025 8:59 PM	Markdown Source ...	1 KB
PROJECT_FEATURES	12/25/2025 8:59 PM	Markdown Source ...	1 KB
README	12/25/2025 8:58 PM	Markdown Source ...	1 KB



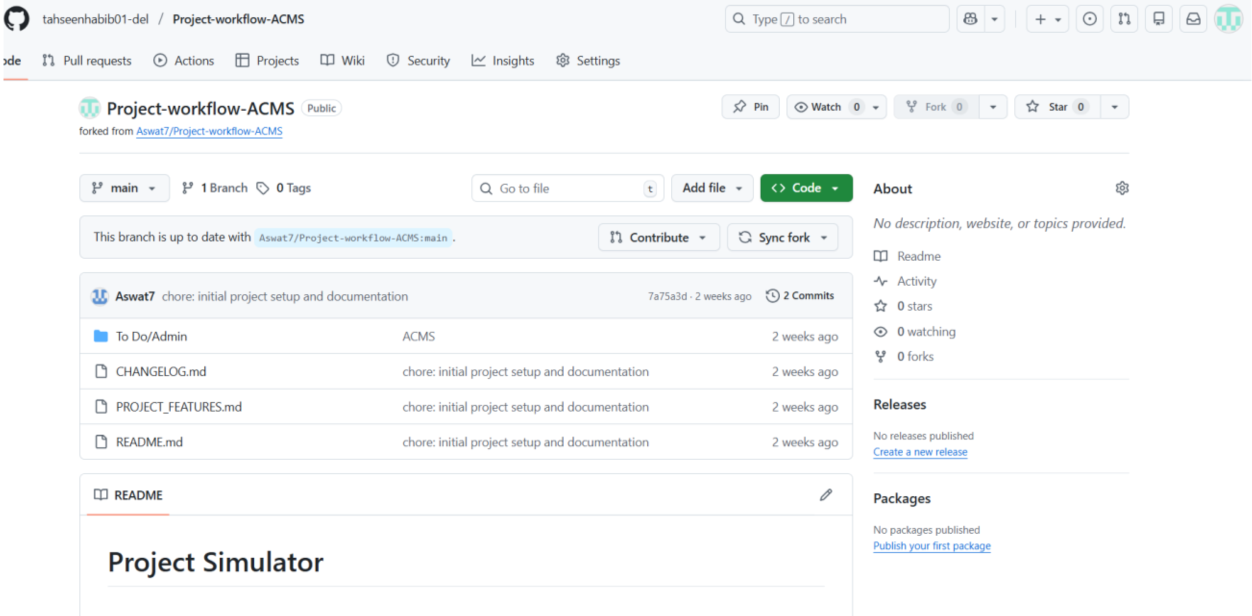
```
Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (dev)
$ git checkout feature
branch 'feature' set up to track 'origin/feature'.
Switched to a new branch 'feature'

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (feature)
$ git remote remove origin

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (feature)
$ git remote add origin https://github.com/Aswat7/Project-workflow-ACMS.git

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (feature)
$ git checkout main
Switched to branch 'main'

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git push -u origin main
remote: Permission to Aswat7/Project-workflow-ACMS.git denied to tahseenhabib01-del.
fatal: unable to access 'https://github.com/Aswat7/Project-workflow-ACMS.git/': The requested URL returned error: 403

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git push -u origin master
error: src refspec master does not match any
error: failed to push some refs to 'https://github.com/Aswat7/Project-workflow-ACMS.git'

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$  git push -u origin main
remote: Permission to Aswat7/Project-workflow-ACMS.git denied to tahseenhabib01-del.
fatal: unable to access 'https://github.com/Aswat7/Project-workflow-ACMS.git/': The requested URL returned error: 403

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git remote remove origin

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ cd ..

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ rm -rf project-worlflow-ACMS

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ git clone https://github.com/Aswat7/Project-workflow-ACMS.git
fatal: destination path 'Project-workflow-ACMS' already exists and is not an empty directory.

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ cd Project-workflow-ACMS

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git branch
  dev
  feature
* main

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git checkout dev
Switched to branch 'dev'

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (dev)
```



```
Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ git config --global user.name "https://github.com/Aswat7/Project-workflow-ACMS.git"

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ git config --global user.email "tahseen.habib01@gmail.com"

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ git clone https://github.com/^C

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$  git clone https://github.com//Aswat7/Project-workflow-ACMS.git
Cloning into 'Project-workflow-ACMS'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 37 (delta 11), reused 27 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (37/37), done.
Resolving deltas: 100% (11/11), done.

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ cd ASWAT7
bash: cd: ASWAT7: No such file or directory

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ cd ASWAT
bash: cd: ASWAT: No such file or directory

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ cd Maliha
bash: cd: Maliha: No such file or directory

Asus@LAPTOP-RVOEVATF MINGW64 ~ (main)
$ cd Project-workflow-ACMS

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git branch -r
  origin/HEAD -> origin/main
  origin/dev
  origin/feature
  origin/main

Asus@LAPTOP-RVOEVATF MINGW64 ~/Project-workflow-ACMS (main)
$ git checkout dev
branch 'dev' set up to track 'origin/dev'.
Switched to a new branch 'dev'
```



```
Asus@LAPTOP-RV0EVATF MINGW64 ~ (main)
$ cd Desktop

Asus@LAPTOP-RV0EVATF MINGW64 ~/Desktop (main)
$ git clone https://github.com/Aswat7/Project-workflow-ACMS.git
Cloning into 'Project-workflow-ACMS'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 37 (delta 11), reused 27 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (37/37), done.
Resolving deltas: 100% (11/11), done.
```



Project-workflow-ACMS Public

forked from Aswat7/Project-workflow-ACMS

This branch is up to date with Aswat7/Project-workflow-ACMS:main .

Aswat7 chore: initial project setup and documentation 7a75a3d · 2 weeks ago 2 Commits

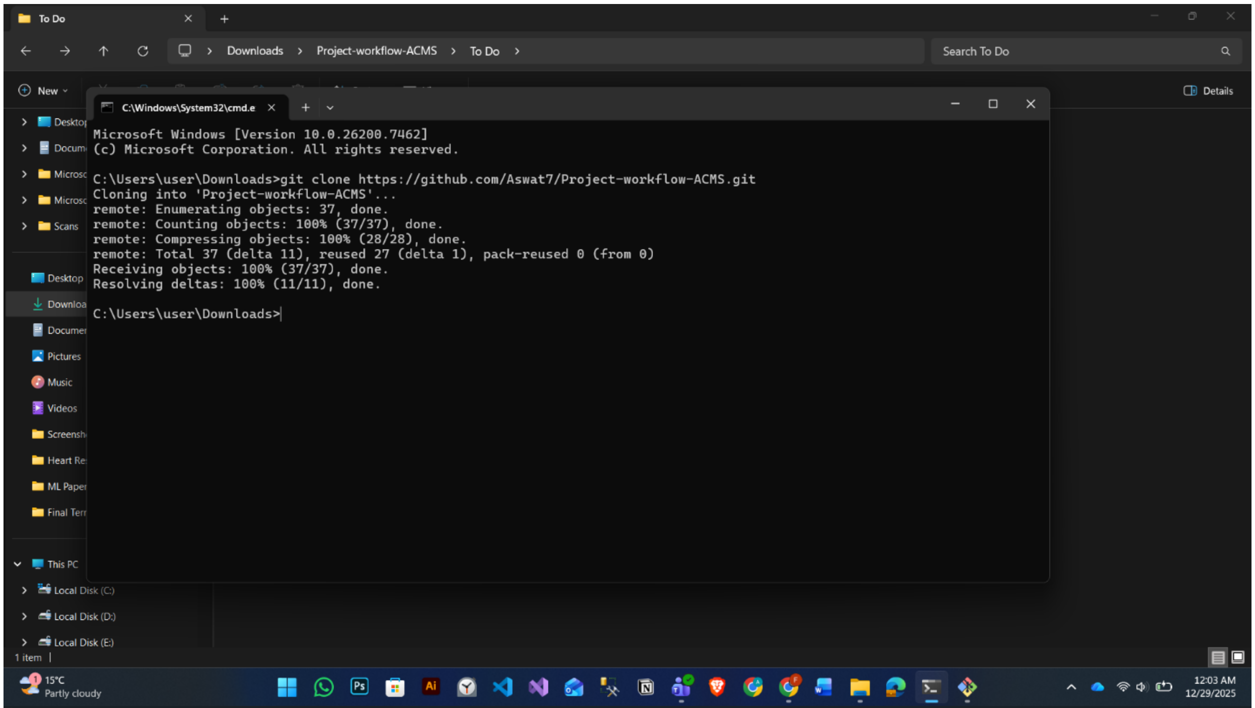
To Do/Admin	ACMS	2 weeks ago
CHANGELOG.md	chore: initial project setup and documentation	2 weeks ago
PROJECT_FEATURES.md	chore: initial project setup and documentation	2 weeks ago
README.md	chore: initial project setup and documentation	2 weeks ago

No description, website, or topics provided.

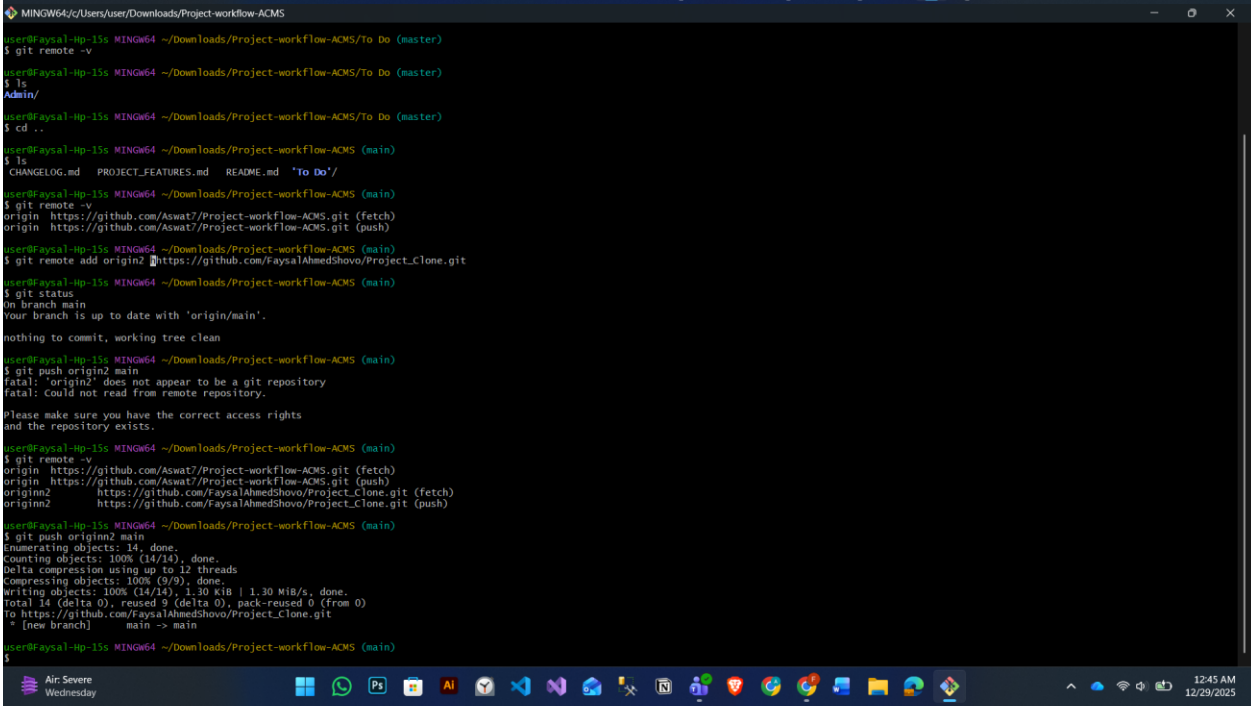
No releases published

No packages published

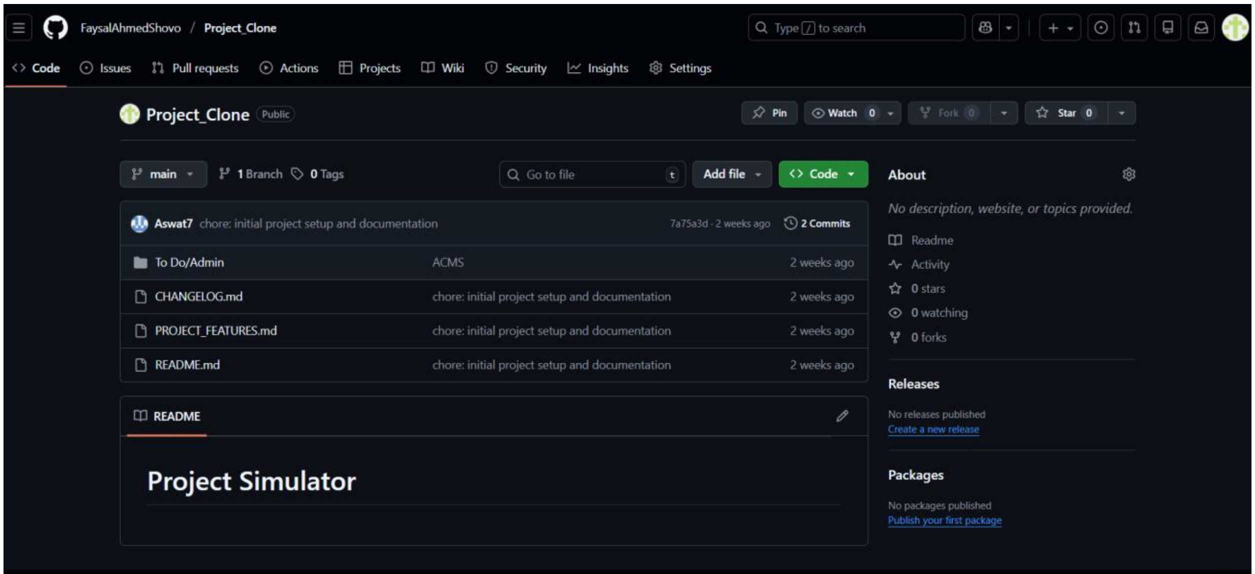
Project Simulator



```
Microsoft Windows [Version 10.0.26200.7462]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user\Downloads>git clone https://github.com/Aswat7/Project-workflow-ACMS.git
Cloning into 'Project-workflow-ACMS'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 37 (delta 11), reused 27 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (37/37), done.
Resolving deltas: 100% (11/11), done.

C:\Users\user\Downloads>
```



```
user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS/To Do (master)
$ git remote -v

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS/To Do (master)
$ ls
Admin/

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS/To Do (master)
$ cd ..

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ ls
 CHANGELOG.md   PROJECT_FEATURES.md   README.md  'To Do'/

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ git remote -v
origin  https://github.com/Aswat7/Project-workflow-ACMS.git (fetch)
origin  https://github.com/Aswat7/Project-workflow-ACMS.git (push)

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ git remote add origin2 https://github.com/FaysalAhmedShovo/Project_Clone.git

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ git push origin2 main
fatal: 'origin2' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ git remote -v
origin  https://github.com/Aswat7/Project-workflow-ACMS.git (fetch)
origin  https://github.com/Aswat7/Project-workflow-ACMS.git (push)
originn2        https://github.com/FaysalAhmedShovo/Project_Clone.git (fetch)
originn2        https://github.com/FaysalAhmedShovo/Project_Clone.git (push)

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$ git push originn2 main
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 12 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (14/14), 1.30 KiB | 1.30 MiB/s, done.
Total 14 (delta 0), reused 9 (delta 0), pack-reused 0 (from 0)
To https://github.com/FaysalAhmedShovo/Project_Clone.git
 * [new branch]      main -> main

user@Faysal-Hp-15s MINGW64 ~/Downloads/Project-workflow-ACMS (main)
$
```



Aswat7 chore: initial project setup and documentation 7a75a3d · 2 weeks ago 2 Commits

To Do/Admin	ACMS	2 weeks ago
CHANGELOG.md	chore: initial project setup and documentation	2 weeks ago
PROJECT_FEATURES.md	chore: initial project setup and documentation	2 weeks ago
README.md	chore: initial project setup and documentation	2 weeks ago

README

Project Simulator

About

No description, website, or topics provided.

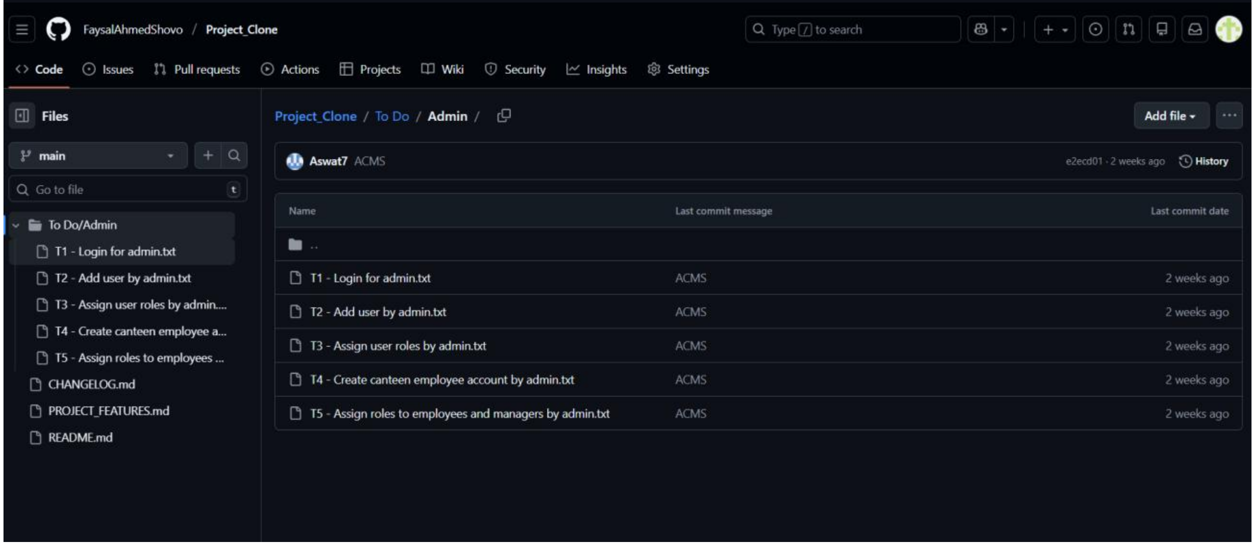
Readme
Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

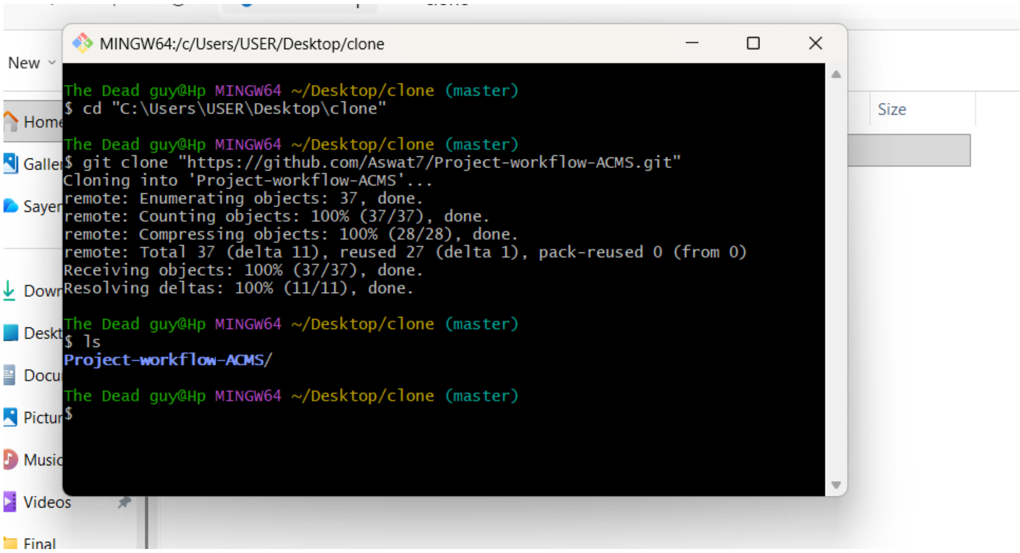
No packages published
Publish your first package



Project_Clone / To Do / Admin /

Aswat7 ACMS e2ecd01 · 2 weeks ago History

Name	Last commit message	Last commit date
..		
T1 - Login for admin.txt	ACMS	2 weeks ago
T2 - Add user by admin.txt	ACMS	2 weeks ago
T3 - Assign user roles by admin.txt	ACMS	2 weeks ago
T4 - Create canteen employee account by admin.txt	ACMS	2 weeks ago
T5 - Assign roles to employees and managers by admin.txt	ACMS	2 weeks ago



```
The Dead guy@Hp MINGW64 ~/Desktop/clone (master)
$ cd "C:\Users\USER\Desktop\clone"

The Dead guy@Hp MINGW64 ~/Desktop/clone (master)
$ git clone "https://github.com/Aswat7/Project-workflow-ACMS.git"
Cloning into 'Project-workflow-ACMS'...
remote: Enumerating objects: 37, done.
remote: Counting objects: 100% (37/37), done.
remote: Compressing objects: 100% (28/28), done.
remote: Total 37 (delta 11), reused 27 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (37/37), done.
Resolving deltas: 100% (11/11), done.

The Dead guy@Hp MINGW64 ~/Desktop/clone (master)
$ ls
Project-workflow-ACMS/

The Dead guy@Hp MINGW64 ~/Desktop/clone (master)
$
```

5. SOFTWARE TESTING

<table>
<tr><td colspan="3">Project Name: AIUB Canteen Management System</td><td colspan="2">Test Designed by: Aswat Shahriar</td></tr>
<tr><td colspan="3">Test Case ID: T1</td><td colspan="2">Test Designed date: 10/12/2025</td></tr>
<tr><td colspan="3">Test Priority (Low, Medium, High): Medium</td><td colspan="2">Test Executed by: Aswat Shahriar</td></tr>
<tr><td colspan="3">Module Name: Login For Admin</td><td colspan="2">Test Execution date: 05/01/2026</td></tr>
<tr><td colspan="3">Test Title: Verify login with valid username and password</td><td colspan="2"></td></tr>
<tr><td colspan="3">Description: Test the website login page</td><td colspan="2"></td></tr>
<tr><td colspan="5">Precondition: The user has a valid username and password</td></tr>
<tr><td colspan="5">Dependences: if any</td></tr>
<tr><td>Test Steps</td><td>Test Data</td><td>Expected Results</td><td>Actual Results</td><td>Status (Pass/Fail)</td></tr>
<tr><td>1. Go to the site
2. Enter username
3. Enter password
4. Click submit</td><td>Username: urs99
Password: 321</td><td>The user should log in to the application</td><td>As expected,</td><td>Pass</td></tr>
</table>

<table>
<tr><td colspan="3">Project Name: AIUB Canteen Management System</td><td colspan="2">Test Designed by: Aswat Shahriar</td></tr>
<tr><td colspan="3">Test Case ID: T2</td><td colspan="2">Test Designed date: 10/12/2025</td></tr>
<tr><td colspan="3">Test Priority (Low, Medium, High): Medium</td><td colspan="2">Test Executed by: Aswat Shahriar</td></tr>
<tr><td colspan="3">Module Name: Add user by admin</td><td colspan="2">Test Execution date: 05/01/2026</td></tr>
<tr><td colspan="3">Test Title: Verify admin can successfully add a new user</td><td colspan="2"></td></tr>
<tr><td colspan="3">Description: Test whether the admin can add a new user by providing valid user details</td><td colspan="2"></td></tr>
<tr><td colspan="5">Precondition: Admin is logged into the system
Admin has permission to add users</td></tr>
<tr><td colspan="5">Dependences: if any</td></tr>
<tr><td>Test Steps</td><td>Test Data</td><td>Expected Results</td><td>Actual Results</td><td>Status (Pass/Fail)</td></tr>
<tr><td>1. Go to the admin dashboard
2. Click on Add User option.
3. Enter valid user details (name, username, password, role, email, etc.)
4. Click submit</td><td>Username: user101
Password: User@123
Role: Employee
Email: user101@gmail.com</td><td>The system should successfully add a new user.
A successful message should be displayed.
The new user should appear in the user list.</td><td>As expected,</td><td>Pass</td></tr>
<tr><td colspan="3"></td><td colspan="2"></td></tr>
</table>

Project Name : AIUB Canteen Management System	Test Designed by: Sayem Mahmud
Test Case ID: T 3	Test Designed date: 20/11/2025
Test Priority (Low, Medium, High): Medium	Test Executed by: Sayem Mahmud
Module Name: Assign user roles by admin	Test Execution date: 28/12/2025
Test Title: Verify that admin can assign roles to a user successfully	
Description: As an admin, I want to assign roles to users so that proper access control is maintained.	
Precondition: Admin is logged into the system	
Dependencies: if any	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Go to the application 2. Login as admin 3. Navigate to User Management 4. Select an existing user 5. Choose a role 6. Save Button	Username: admin44 Password:@33455	1.The selected role is successfully assigned to the user 2. System shows a confirmation message 3. User role is updated in the system	As expected	Pass

Project Name : AIUB Canteen Management System	Test Designed by: Sayem Mahmud
Test Case ID: T 4	Test Designed date: 20/11/2025
Test Priority (Low, Medium, High): Medium	Test Executed by: Sayem Mahmud
Module Name: Create canteen employee account by admin	Test Execution date: 28/12/2025
Test Title: Verify that admin can create a canteen employee account successfully	
Description: This test case verifies that the admin can create a new canteen employee account by providing valid employee details.	
Precondition: Admin is logged into the system Admin has access to the employee creation module Required employee information is available	
Dependencies: if any	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
5. Go to the application 6. Login as admin 7. Navigate to Employee Management 8. Click Add New Employee 9. Enter employee details 10. Save Button	Username: Karim Hasan Password:@3342	1. New employee account is created successfully 2. System displays a success message 3. Employee appears in the employee list	As expected	Pass

Project Name : AIUB Canteen Management System	Test Designed by: Faysal Ahmed Shovo
Test Case ID: T5	Test Designed date: 21/11/2025
Test Priority (Low, Medium, High): Medium	Test Executed by: Faysal Ahmed Shovo
Module Name: Assign roles to employees and managers by admin	Test Execution date: 29/12/2025
Test Title: Verify that the admin can successfully assign roles to employees and managers.	
Description: The admin can assign and update roles for employees and managers accurately, and that the system enforces the correct access permissions based on the assigned roles.	
Precondition: 1.Admin user is logged into the system 2. Employees and managers are already registered in the system 3. Admin has permission to manage user roles	
Dependencies: if any	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Log in to the system using valid admin credentials. 2. Navigate to the User Management / Role Management module. 3. Select an existing employee or manager from the user list. 4. Choose a role (Employee or Manager) from the available role options. 5. Click on the Assign / Save button. 6. Log out from the admin account. 7. Log in using the assigned user's credentials.	Username: admin33 Password: 321@	1. The selected role is successfully assigned to the user. 2. A confirmation message is displayed after role assignment. 3. The user gains access only to features permitted for their assigned role.	As expected	Pass

Project Name : AIUB Canteen Management System	Test Designed by: Faysal Ahmed Shovo
Test Case ID: T6	Test Designed date: 21/11/2025
Test Priority (Low, Medium, High): Medium	Test Executed by: Faysal Ahmed Shovo
Module Name: Registration for User	Test Execution date: 29/12/2025
Test Title: Verify that a new user can successfully register using valid details.	
Description: Test whether a new user can successfully register using valid name, ID, and university email.	
Precondition: 1. User is not already registered in the system 2. User has a valid AIUB university email address 3. Registration page is accessible	
Dependencies: if any	

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1. Open the AIUB Canteen Management System website/application 2. Navigate to the User Registration page 3. Enter valid user details (Name, ID, University Email) 4. Click on the Register / Sign Up button 5. Submit the registration form	Username: Shovo33 Password: 321@	1. User registration is completed successfully 2. A confirmation message is displayed 3. User account is created and stored in the system 4. User is able to log in using the registered credentials	As expected	Pass

Project Name : AIUB Canteen Management System	Test Designed by: Tahseen Habib Maliha
Test Case ID: T7	Test Designed date: 10/12/2025
Test Priority (Low, Medium, High): Medium	Test Executed by: Tahseen Habib Maliha
Module Name: Secure login for user	Test Execution date: 05/01/2026
Test Title: Verify secure login for user with valid credentials	
Description: Test whether a registered user can securely log in using a valid username and password	

Precondition:

- User account exists and is active
- Users have valid login credentials

Dependencies: if any

Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1 Go to the login page 2 Enter valid username 3 Enter valid password 4 Click on the Login button	• Username: user99 • Password: User@321	1. User should be logged into the system securely 2. User dashboard should be displayed 3. No unauthorized access should be allowed	As expected	Pass

Project Name : AIUB Canteen Management System			Test Designed by: Tahseen Habib Maliha	
Test Case ID: T7			Test Designed date: 10/12/2025	
Test Priority (Low, Medium, High): Medium			Test Executed by: Tahseen Habib Maliha	
Module Name: Canteen Manager Login			Test Execution date: 05/01/2026	
Test Title: Verify login functionality for canteen manager				
Description: Test whether the canteen manager can log in using valid credentials				
Precondition: • Canteen manager account exists and is active • Valid login credentials are available				
Dependencies: if any				
Test Steps	Test Data	Expected Results	Actual Results	Status (Pass/Fail)
1 Navigate to the login page 2 Enter canteen manager username 3 Enter canteen manager password 4 Click on the Login button	• Username: manager01 • Password: Manager@123	1. Canteen manager should successfully log in 2. Manager dashboard should be displayed 3. Access should be limited to manager privileges	As expected	Pass

6. CONCLUSION

The AIUB Canteen Management System (ACMS) is designed to modernize and automate the traditional manual operations of the university canteen. The system addresses critical issues such as long queues, slow order processing, miscommunication, and inefficient record management by introducing a centralized digital platform. Through features like online ordering, digital payment, real-time order tracking, feedback management, and automated reporting, ACMS significantly improves operational efficiency and service quality.

This project follows the Agile Scrum methodology, allowing continuous feedback, incremental development, and adaptability to changing requirements. The system supports multiple user roles, including students, faculty members, staff, canteen employees, managers, and administrators, ensuring role-based access and secure system usage. Both functional and non-functional requirements have been carefully analyzed to ensure reliability, performance, availability, and user satisfaction.

Comprehensive testing confirms that the core functionalities perform as expected, meeting system requirements effectively. Overall, ACMS provides a scalable, user-friendly, and efficient solution that enhances the canteen experience while reducing administrative burden. The successful implementation of this system demonstrates how software engineering principles can be applied to solve real-world problems in an academic environment.