

Uwaterloo Exchange by Group 15

The student hub for buying, selling, and Swapping Essentials



UNIVERSITY OF
WATERLOO

| FACULTY OF
ENGINEERING

Why we wanted to make this app?

The Waterloo Exchange app was created to address the need for students to easily buy, sell, and exchange items within the Waterloo community. With features tailored for students, it provides a dedicated platform to conveniently list and find textbooks, electronics, clothing, and other items.

- Verified users
- Nearby exchange
- Dedicated platform
- Scam less likely



UNIVERSITY OF
WATERLOO

| FACULTY OF
ENGINEERING

Key Features

- Post management

Create, browse, update, and delete listings.

Bookmark / remove a listing for a customer

Own listing page

Mark Order as completed for completed transaction

Track Completed Order with a separate listing page

- Customer profile management

Create, read, update, and delete customer profiles.

- Authentication

Incorporate authentication into our software.

- Chat

Enable chat functionality between customers.

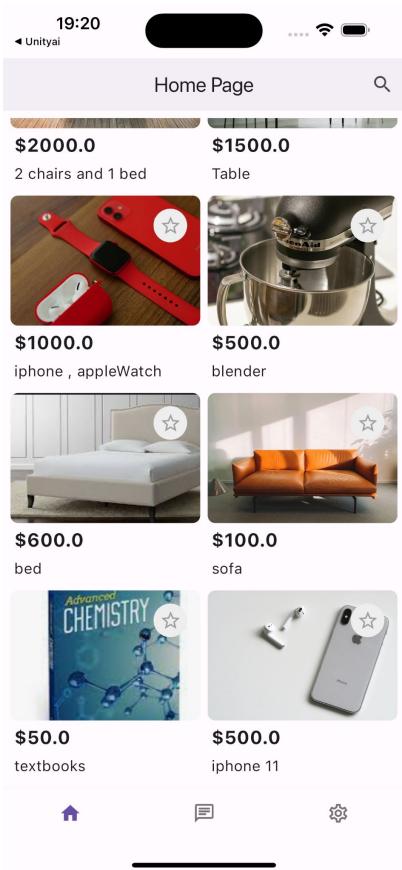
Directly Send Offer to other user



UNIVERSITY OF
WATERLOO

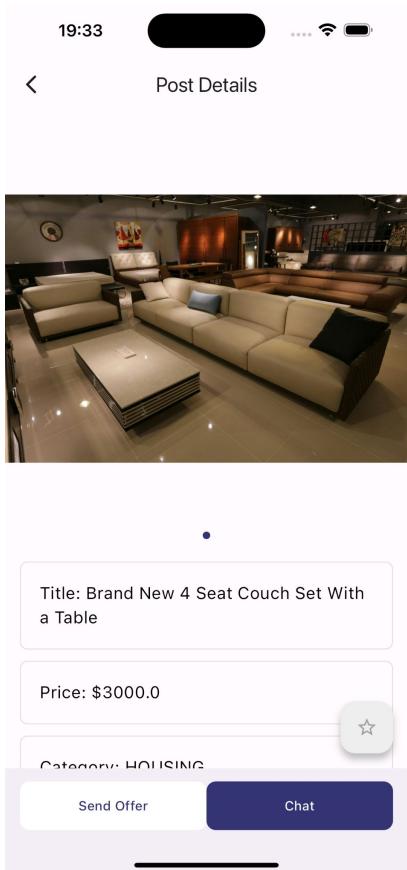
| FACULTY OF
ENGINEERING

User Interface



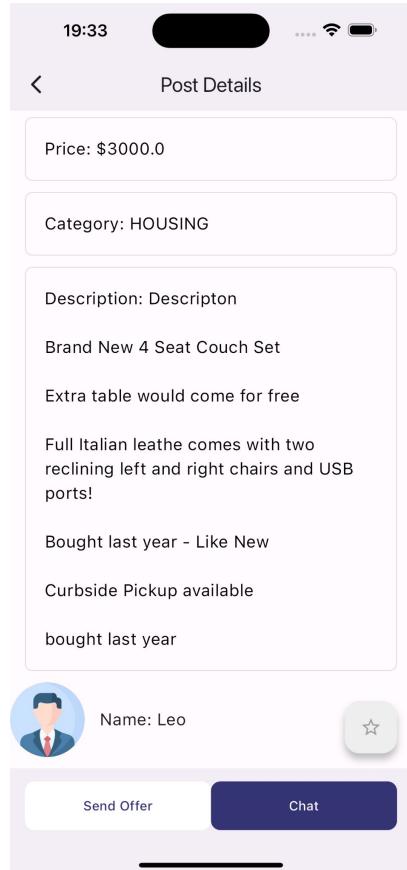
Main Page

People can check for available listings



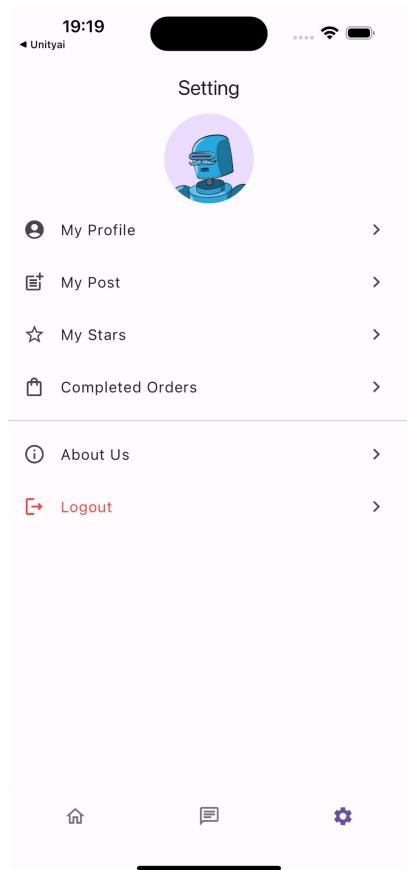
Listing Details

People may expand a listing



Listing Details

People may check for additional details



Customer Profile

People update their profile and manage their own listings



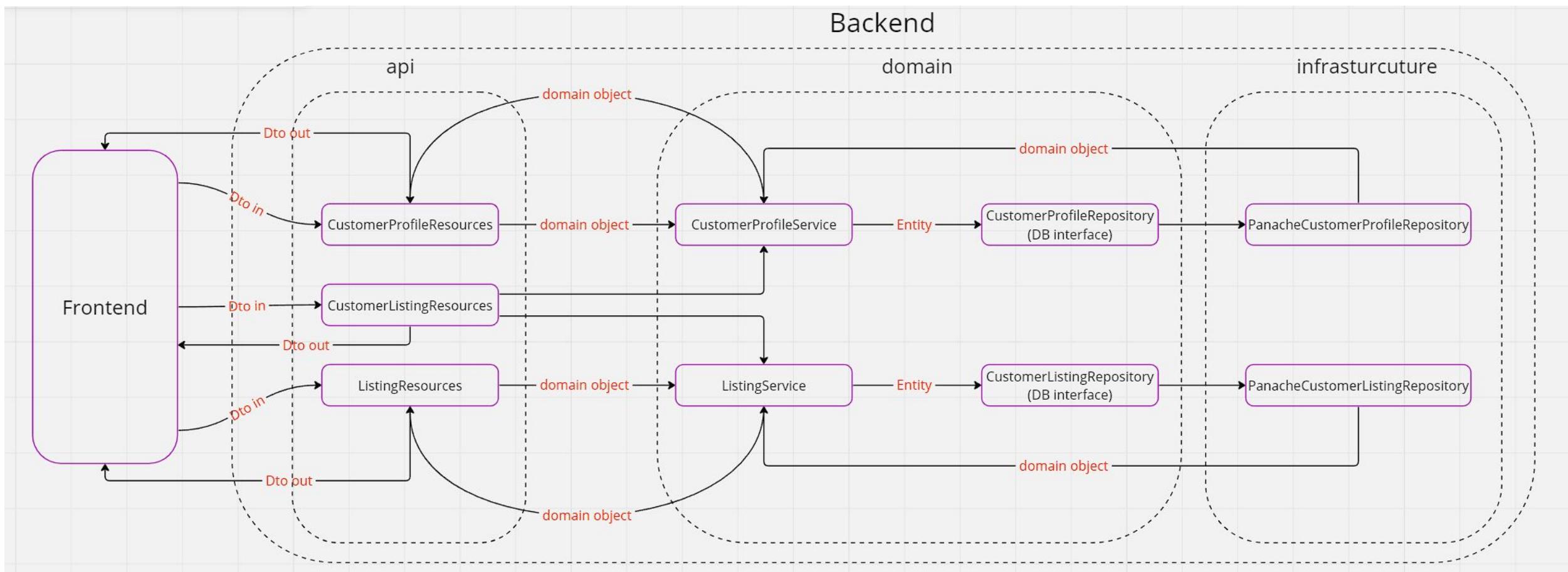
Real Time Chat Functionality



UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

Project Architecture



UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

Practices Followed – Agile

user stories and features are broken into tasks.

We used **kanban** board to track frontend and backend issues/tasks:

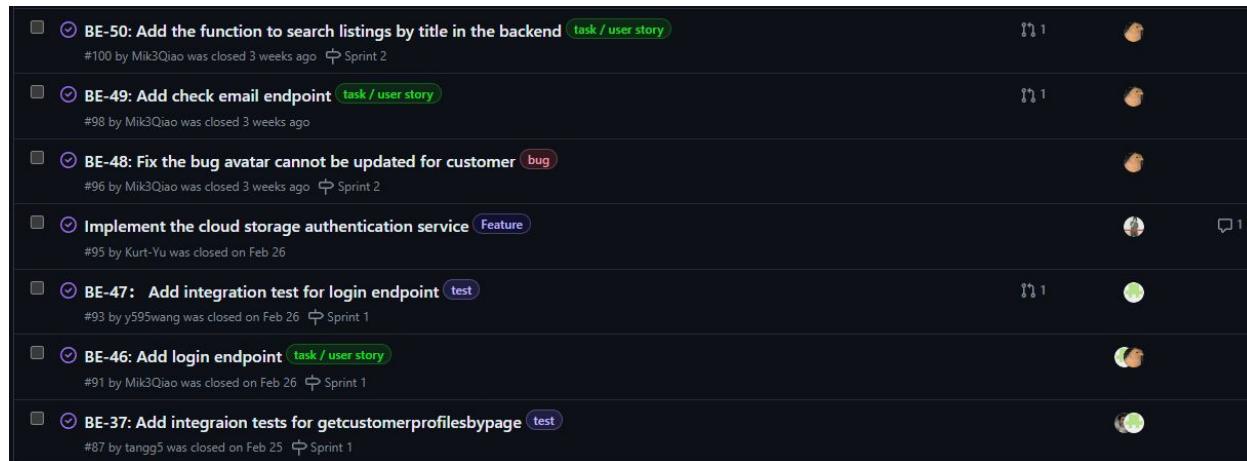
- TODO
 - Something need to be considered for next sprint.
- In progress:
 - Should have tasks for the current sprint
- In Review with a maximum of 5 tasks:
 - Promotes code review within the team
- Done

The screenshot shows a digital kanban board interface with the following columns and items:

- No Status**: Contains 4 items (Estimate: 0).
 - backend #108: BE-54: Add unit tests for ListingService
 - backend #111: BE-56: Add integration tests for get completed listings for a customer
 - backend #123: Be 60 Add Integration tests for ListingResource
 - backend #124: BE-62: Add IT for CustomerProfileResources
- Todo**: Contains 2 items (Estimate: 0).
 - Draft: Feature 0: Authentication and Authorization: our application can verify user identity
- In progress**: Contains 0/3 items (Estimate: 0).
 - This item is actively being worked on
- In review**: Contains 0/5 items (Estimate: 0).
 - This item is in review
- Done**: Contains 195 items (Estimate: 97).
 - This item has been completed.
 - mobile #12: MB-4: Add the function to upload user-selected images to the cloud and return links to backend (P0 L)
 - Draft: FF: skeleton setup and code structure organization (P0 10)
 - mobile #14: MB-11: Post detail page (P0 L)
 - backend #96: BE-48: Fix the bug avatar cannot be updated for customer (P0 1 XS)
 - backend #91: BE-46: Add login endpoint (P0 2 S)
 - mobile #21: MB-8: login interface with backend api (P0 L)

Practices Followed – Issues

- Each Feature is divided into user stories, and each user story is then used to create an issue
- We have a numbering system for the issues, for example this one is BE-37, which helps us to track commits related to specific issue
- Issues are then tagged with different categories: **task/user story, test, documentation, bug etc.**



BE-50: Add the function to search listings by title in the backend (task / user story)
#100 by Mik3Qiao was closed 3 weeks ago → Sprint 2

BE-49: Add check email endpoint (task / user story)
#98 by Mik3Qiao was closed 3 weeks ago

BE-48: Fix the bug avatar cannot be updated for customer (bug)
#96 by Mik3Qiao was closed 3 weeks ago → Sprint 2

Implement the cloud storage authentication service (Feature)
#95 by Kurt-Yu was closed on Feb 26

BE-47: Add integration test for login endpoint (test)
#93 by y595wang was closed on Feb 26 → Sprint 1

BE-46: Add login endpoint (task / user story)
#91 by Mik3Qiao was closed on Feb 26 → Sprint 1

BE-37: Add integraion tests for getcustomerprofilesbypage (test)
#87 by tangg5 was closed on Feb 25 → Sprint 1



BE-37: Fix the bug when created a listing, calling get posting listing returns exception. #58

Closed Mik3Qiao opened this issue on Feb 18 · 2 comments

Mik3Qiao commented on Feb 18 · edited As discussion during the meeting, this bug was spotted by the FE folk @Kurt-Yu, created this ticket to fix the issue when create a new customer, when calling get their posted listing endpoint, returns exception.

Mik3Qiao added this to UW exchange on Feb 18

Mik3Qiao converted this from a draft issue on Feb 18

Mik3Qiao self-assigned this on Feb 18

Mik3Qiao added the bug label on Feb 18

Mik3Qiao added this to the Sprint 1 milestone on Feb 18

tangg5 commented on Feb 18 I checked the getCustomerPostedListings function and the message format printed, it doesn't seem to be throwing exceptions, the format is not defined by the exception mapper

Mik3Qiao commented on Feb 18 Yeah this is related to the toDomain method in CustomerProfile dto

Mik3Qiao moved this from In progress to In review in UW exchange on Feb 19

Mik3Qiao moved this from In review to Done in UW exchange on Feb 19

Practices Followed – Sprint Planning: Planning Poker

For example this specific MB ticket: Mobile

- We have a numbering system for the issues: **MB-14** is included in commit message, easy to tell which commit is related to which task
- During the team meeting, we use planning poker to point the tickets:
 - For example this one is **P1** means it's less important than **P0, P0** might be something like getting my postings
 - 2 points means the effort level for this function is **medium-low**
- This task is then moved to **Sprint 1**

The screenshot shows a GitHub issue page for a ticket titled "MB-14: add delete button on mypost page #27". The ticket is marked as "Closed" and has a status of "Public". The description reads: "As a user, I want to be able to delete my posting on my postings page." The right sidebar displays the following details for the project "UW exchange":

Status	Done
Priority	P1
Size	M
Estimate	2
Start date	No date
End date	No date

The sidebar also shows the ticket is assigned to "yunmark" and is labeled as a "task/user story". It is linked to the "Sprint 1" milestone, which is due on Feb 26 and 100% complete. A pull request has been linked to close the issue. The ticket was moved from Todo to In review by "yunmark" on Feb 24. It was closed as completed by "leozlt0112" on Feb 24. The ticket was moved from In review to Done by "github-project-automation" on Feb 24.

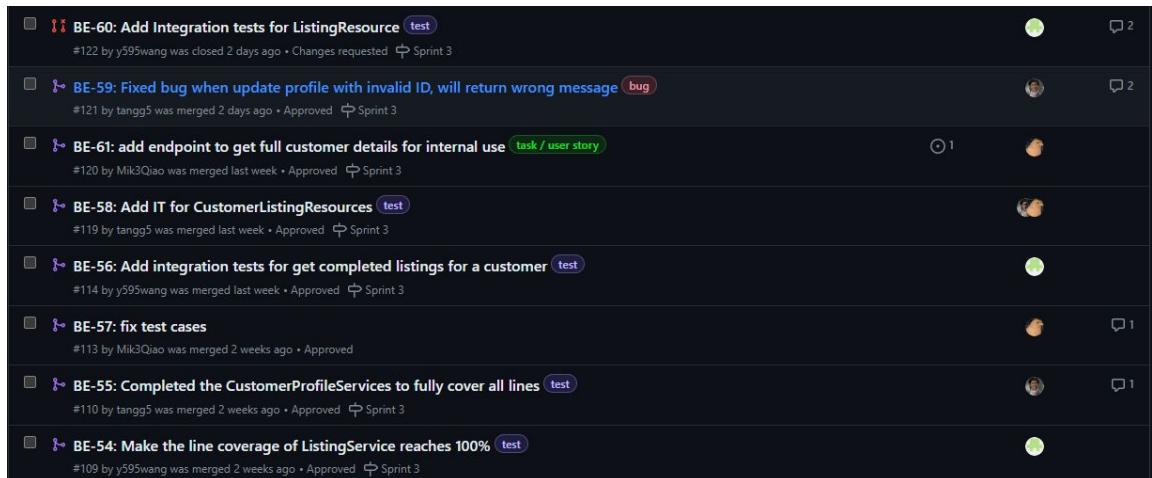


UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

Practices Followed – PRs

- We have a PR template which includes necessary information needed.
- Author will need to provide a brief description of the changes made
- Main branch is protected.
 - At least one Team Member from the group will need to review the changes in order to be merged to main



BE-12: Code reformat and add relation for profile and listing #20

Merged tangg5 merged 2 commits into main from BE-12 on Feb 13

Description

- Describe the changes in this Pull Request, and related ticket if applicable.
 - In this ticket, code was reformatted so that resource layer has simpler logic which is basically telling the value returned from service layer, and service layer doing all the business logic. Persistence layer is responsible for persisting data and update data.
 - Customer starred listing has a many to many relation with Listing.
 - Customer owned listing has a one to many relation with Listing.

Tests

- How did you test this change? Or how can someone test it?
 - Pass local tests, and works in postman.
 - Now if a customer profile does not exist, they cannot create a listing.

Screenshots

If applicable, attach screenshots to explain the changes.

BE-12: Fix the conflicts and adjust existing code ...

Notifications

Unsubscribe

You're receiving notifications because you modified the open/close state.

2 participants

Lock conversation

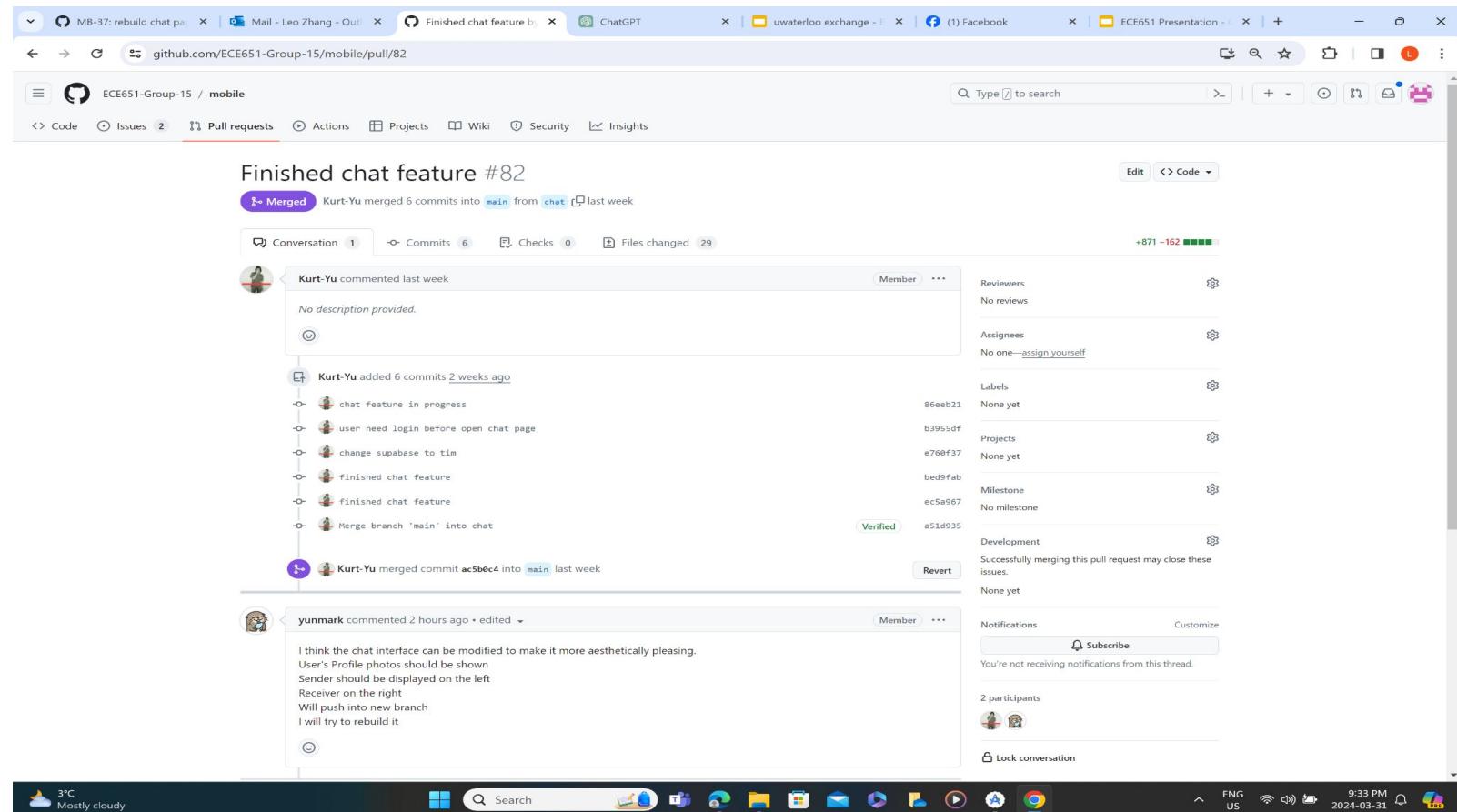


UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

PRs Continued

Code Review Example



Practices Followed – CI/CD

- Realized via:
 - Github action
 - Github Secrets which stores the **Sensitive information** that was **intentionally** hidden from the user
 - AWS EC2 instance which runs our application

All workflows			
Showing runs from all workflows			
Event	Status	Branch	Actor
2 days ago	Success	main	Mik3Qiao
last week	Success	main	Mik3Qiao
last week	Success	main	tangg5
last week	Success	main	tangg5
2 weeks ago	Success	main	y595wang
2 weeks ago	Success	main	y595wang
2 weeks ago	Success	main	tangg5
2 weeks ago	Success	main	tangg5
3 weeks ago	Success	main	Mik3Qiao
3 weeks ago	Success	main	tangg5
3 weeks ago	Success	main	tangg5
3 weeks ago	Success	main	tangg5
3 weeks ago	Success	main	Mik3Qiao
3 weeks ago	Success	main	tangg5



UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

Practices Followed – CI/CD

- Github actions
 - It is triggered every time a PR got merged to **main**
 - When a PR get merged to main:
 - Github action initialize the connection to our AWS EC2 instance
 - Pull the changes and build **while running all existing tests**
 - Replace existing running instance

Practices Followed – Team Meetings

- Online Meeting – MS Teams – Every Tuesday and Sunday
- Ad-hoc Meetings scheduled throughout the term
 - Online & in-person
 - Depends on the nature of the issue
- Topics and contents involved will be recorded in the meeting notes
- All members are actively engaged

Meeting Notes
Guangwei Tang edited this page last week - 8 revisions Edit New page

Sprint 1 / Milestone 1:

Meeting 1
Jan 14:

- Agenda: Discuss project ideas and finalize our project scope
- Notes:
 - We explored some options of the project. Everyone needed to come up with 5 ideas. Since many students proposed ideas related to social media and item selling, the team then formed a consensus to build an online exchange platform, Uwaterloo Exchange.
 - The team asked each member to think about what area of the project they want to work on, (frontend/backend) Teammates are encouraged to work on an area based on interest and preferences.
 - Teammates are encouraged to express their ideas/questions via online chat. They are also encouraged to raise concerns before and during meetings.

Meeting 2
Jan 16

- Agenda: Setup github repository and divide tasks into each group members
- Notes:
 - Our project is a mobile app that requires both frontend user interaction and backend services, and we have six members in our group, so we decided that three people work
 - Front-end(FE) and Back-end(BE)
 - on the backend APIs and three people work on mobile UI implementation
 - Yu Xia, Leo Zhang, Ziniu Yu - FE | Hualong Qia Guangwei Tang Yue Wang BE
 - We finalized on the framework / technology used for FE and BE, also setup the github repo and committed the basic code skeleton
 - Teammates are encouraged to think what tasks are needed

Meeting 3
Jan 21

- Agenda: Weekly updates about project progress
- Notes:
 - We setup the kanban board on github and divide our user requirements into small user stories and trackable issues
 - Came up with three milestones:
 - The first milestone is to implement post-related API services, such as create post, show post lists, update post and delete post, also part of profile related service.

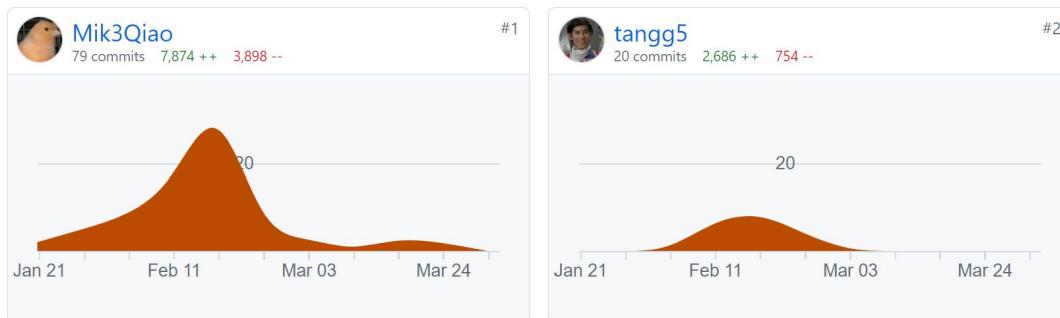


UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

Practices Followed – Contributions

All group members were actively engaged and fairly contributed



Practices Followed – Testing

Backend using onion architecture:

- Resources (api)
- Services
- Domain

Line coverage for all Apis, services, and repositories (db operations) are 100%
Junit, rest assured for IT

Element	Class, %	Method, %	Line, %
all	85% (80/94)	88% (281/323)	93% (997/1066)
infrastructure	80% (55/68)	83% (167/199)	91% (604/659)
Main	0% (0/1)	0% (0/5)	0% (0/11)
CustomerProfileResources	100% (1/1)	100% (5/5)	100% (49/48)
CustomerListingResources	100% (1/1)	100% (8/8)	100% (85/85)
ListingResources	100% (1/1)	100% (11/11)	100% (121/121)
sql	83% (5/6)	70% (35/50)	85% (137/160)
result	100% (16/16)	100% (45/45)	100% (45/45)
dto	75% (3/4)	85% (63/74)	89% (168/188)
domain	96% (25/26)	91% (14/14)	96% (393/407)
MDSUtil	100% (1/1)	100% (2/2)	66% (8/12)
profile	100% (9/9)	91% (45/49)	97% (177/181)
UpdateCustomerProfile	100% (2/2)	81% (9/11)	81% (9/11)
Login	100% (2/2)	80% (4/5)	80% (4/5)
CustomerProfileRepository	100% (1/1)	100% (11/11)	100% (143/143)
CustomerProfile	100% (2/2)	100% (13/13)	100% (13/13)
CreateCustomerProfile	100% (2/2)	88% (8/9)	88% (8/9)
listing	93% (15/16)	91% (67/73)	97% (208/214)
UpdateListing	100% (2/2)	92% (12/13)	92% (12/13)
StarListing	100% (2/2)	80% (4/5)	80% (4/5)
SearchListing	100% (2/2)	75% (3/4)	75% (3/4)
PostedListingPage	0% (0/1)	0% (0/2)	0% (0/2)
ListingWithCustomerInfo	100% (2/2)	100% (3/3)	100% (3/3)
ListingStatus	100% (1/1)	100% (2/2)	100% (3/3)
ListingService	100% (1/1)	100% (12/12)	100% (147/147)
ListingRepository	100% (0/0)	100% (0/0)	100% (0/0)
ListingDetails	100% (2/2)	100% (16/16)	100% (16/16)
CreateListing	100% (2/2)	92% (13/14)	92% (13/14)
Category	100% (1/1)	100% (2/2)	100% (7/7)



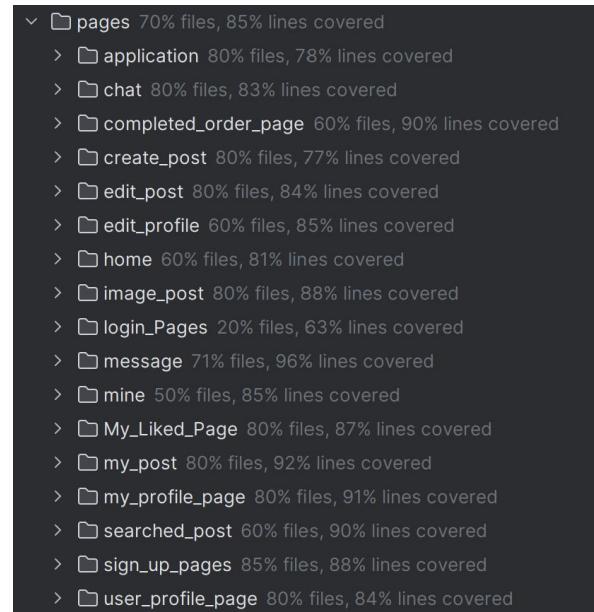
UNIVERSITY OF
WATERLOO

FACULTY OF
ENGINEERING

Practices Followed – Testing Continued

Frontend using flutter integration test architecture

- Simulated automated UI testing

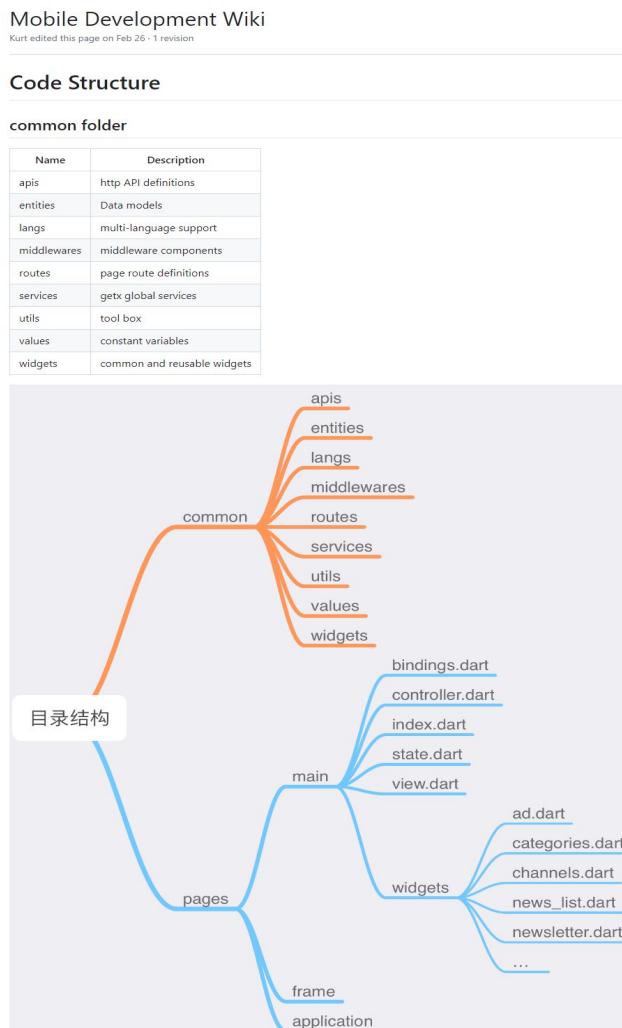


LCOV Viewer					
74.8% Files: 2074/2771 100.0% Functions: 0/0 100.0% Branches: 0/0					
File Lines Line Coverage Functions Function Coverage Branches Branch Coverage					
- All Files	2074/2771	74.8%	0/0	100.0%	0/0 100.0%
- lib	2074/2771	74.8%	0/0	100.0%	0/0 100.0%
main.dart	12/15	80.0%	0/0	100.0%	0/0 100.0%
- pages	1544/1830	84.4%	0/0	100.0%	0/0 100.0%
- edit_profile	76/69	85.4%	0/0	100.0%	0/0 100.0%
state.dart	7/8	87.5%	0/0	100.0%	0/0 100.0%
bindings.dart	0/2	0.0%	0/0	100.0%	0/0 100.0%
controller.dart	41/47	87.2%	0/0	100.0%	0/0 100.0%
view.dart	28/32	87.5%	0/0	100.0%	0/0 100.0%
- application	33/42	78.6%	0/0	100.0%	0/0 100.0%
bindings.dart	5/5	100.0%	0/0	100.0%	0/0 100.0%
view.dart	17/17	100.0%	0/0	100.0%	0/0 100.0%
controller.dart	10/18	55.6%	0/0	100.0%	0/0 100.0%
state.dart	1/2	50.0%	0/0	100.0%	0/0 100.0%
- create_post	127/164	77.4%	0/0	100.0%	0/0 100.0%
bindings.dart	2/2	100.0%	0/0	100.0%	0/0 100.0%
view.dart	87/109	79.8%	0/0	100.0%	0/0 100.0%
controller.dart	35/49	71.4%	0/0	100.0%	0/0 100.0%
state.dart	3/4	75.0%	0/0	100.0%	0/0 100.0%
- image_post	141/160	88.1%	0/0	100.0%	0/0 100.0%
bindings.dart	2/2	100.0%	0/0	100.0%	0/0 100.0%

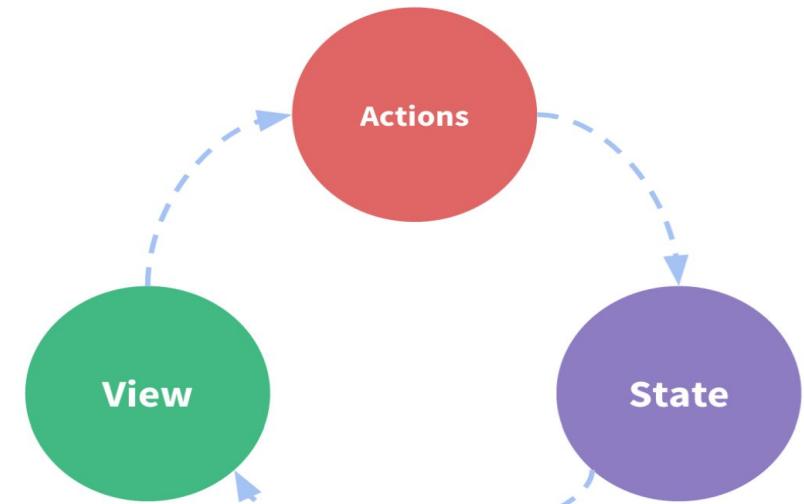
Practices Followed – Frontend Standard

MVC principle

- increase readability
- provides modularity



pages folder



Each page has the following files, based on the philosophy of redux, we divide the code into view/state/action and make our structure more readable and improves the modularity.

Name	Description
bindings.dart	data bindings
controller.dart	controller
index.dart	entrance of each page
state.dart	state variables
view.dart	view
widgets	widget components



UNIVERSITY OF
WATERLOO

| FACULTY OF
ENGINEERING

Tech Stacks



Cross Platform mobile App

The frontend is built with Dart which provides high performance mobile UI. Runs on Android, iOS, Web, Windows, MacOS & Linux



Java Backend

Backend built with Java, Quarkus, Panache & Postgres for scalability & performance, lombok to simplify codebase. Also use onion architecture for our project



PostgreSQL database with Panache ORM

The app uses PostgreSQL for data storage and Panache for simplified database interactions.



CI/CD pipelines with GitHub Actions

The app uses GitHub Actions workflows for continuous integration and deployment to ensure quick and reliable updates.



Hosted on AWS EC2

The app backend and database runs on Amazon Web Services for scalability and reliability.



Unit & Integration Testing

JUnit 5, REST Assured to enable test driven development

The app uses a modern tech stack optimized for performance, scalability and efficient development.

Cross-platform

ndroi



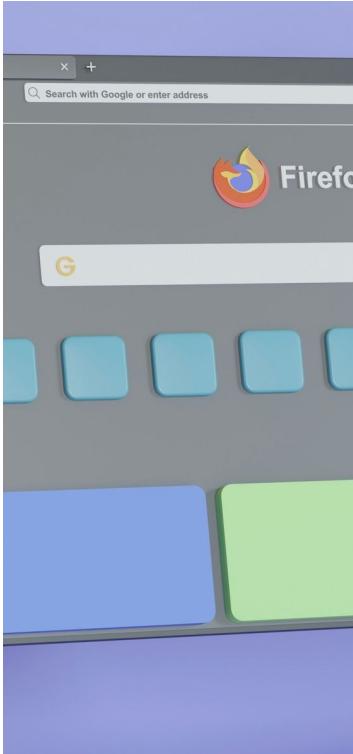
Android

The app can run on
Android devices.



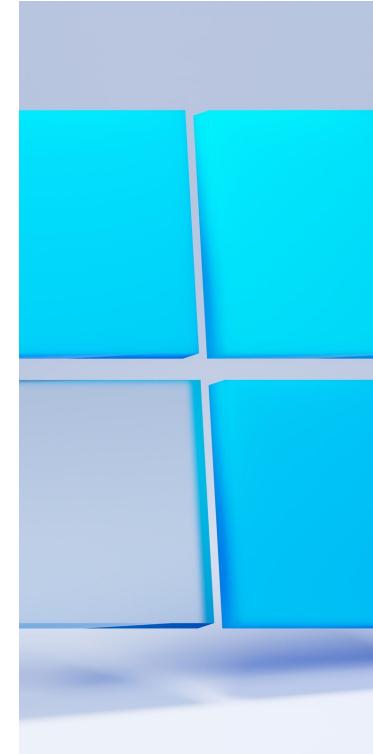
Apple

The app can run on iOS
devices like iPhone and
iPad.



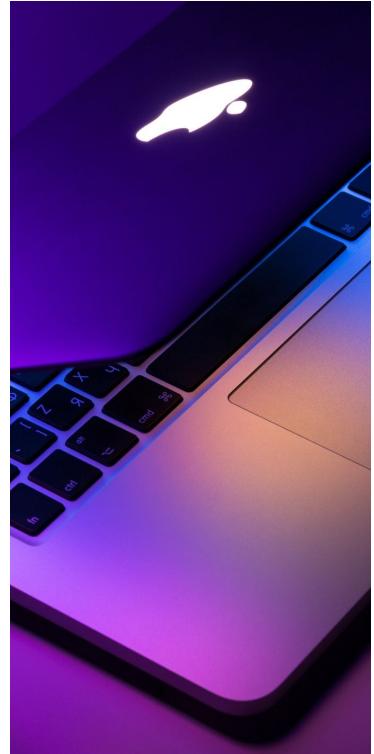
Web browser

The app can run on web
browsers like Chrome,
Firefox and Safari.



Windows

The app can run on
Windows PCs and
tablets.



MacOS

The app can run on Mac
computers.



Linux

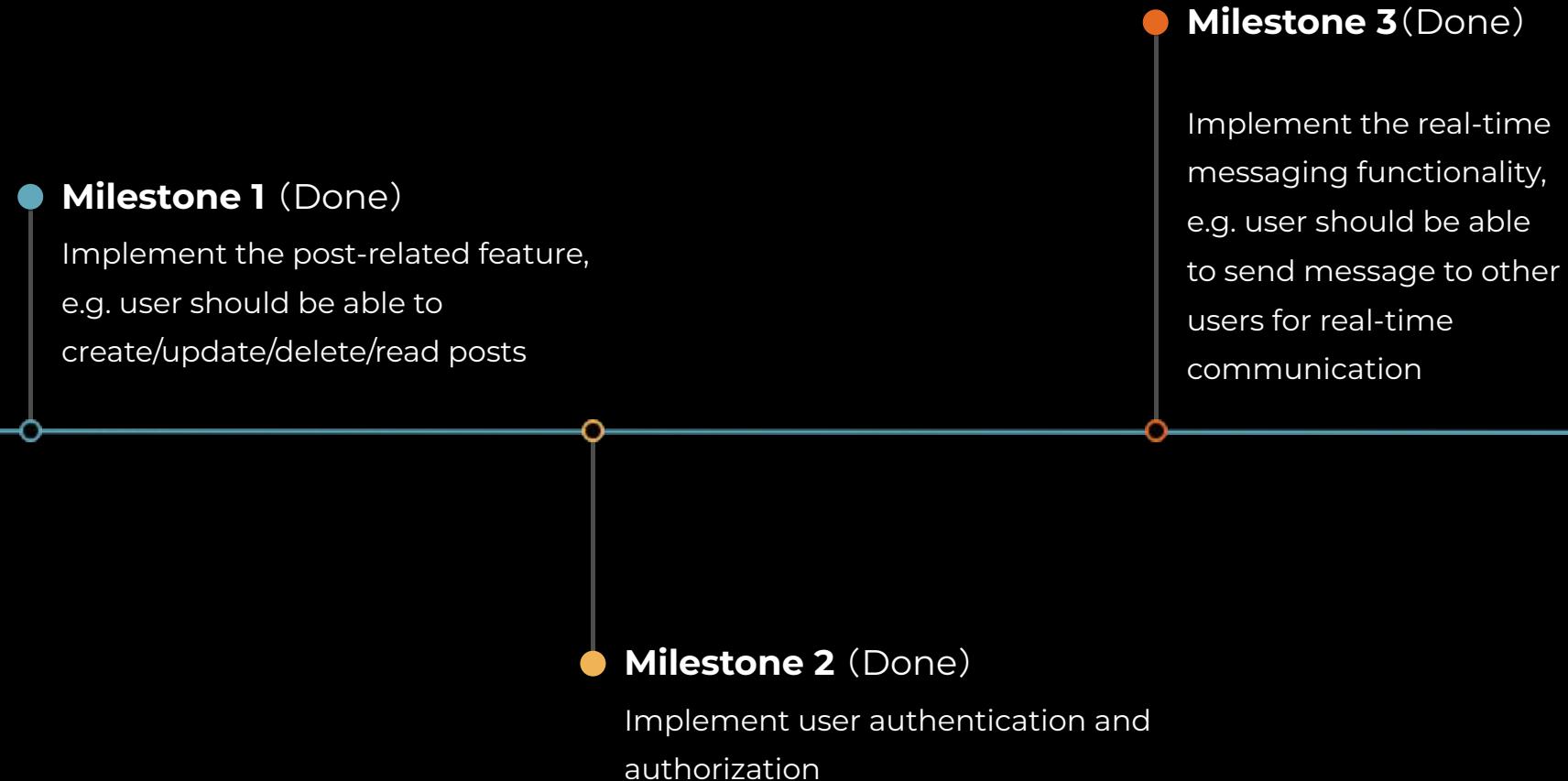
The app can run on
Linux operating
systems.



UNIVERSITY OF
WATERLOO

| FACULTY OF
ENGINEERING

Project Milestones



UNIVERSITY OF
WATERLOO

| FACULTY OF
ENGINEERING

Project Management

- **Project feature and user story breakdown:**

<https://github.com/orgs/ECE651-Group-15/projects/2>

- **Backend issue tracking:**

<https://github.com/ECE651-Group-15/backend/issues>

- **Frontend issue tracking:**

<https://github.com/ECE651-Group-15/mobile/issues>

- **Kanban Board:**

<https://github.com/orgs/ECE651-Group-15/projects/2/views/1>

- **Weekly meeting records:**

<https://github.com/ECE651-Group-15/backend/wiki/Meeting-Notes>



UNIVERSITY OF
WATERLOO

| FACULTY OF
ENGINEERING