The TEZ App (Deliverable 3)

Prepared by Muaz Ahmed(22i-1125), Ibtehaj Haider(22i-0767), Usman Haroon(22i-1177)

TEZ

29/04/2025

Table of Contents

1. Introduction

1.1 Purpose

The TEZ app is designed to modernize and enhance bus transportation management for educational institutions, businesses, and public transport companies in Pakistan. It provides a centralized platform that allows administrators, drivers, and passengers to coordinate, track, and manage bus services more efficiently.

This document outlines the software requirements for the TEZ app, detailing its features to plan the route, real-time seat viewing, live bus tracking, scheduling, and maintenance. By ensuring accurate, real-time information flow between users, drivers, and administrators, the app aims to improve user experience, organization efficiency, avoiding miscommunication, and ensure the effective utilization of transport resources. It addresses a growing need in Pakistan, where no comprehensive transport management system currently exists, while drawing inspiration from successful international solutions such as Singapore's metro system.

1.2 Product Scope

Key features include real-time seat availability, live bus tracking, maintenance scheduling, emergency notifications, and online payment handling as well as virtual bus passes.
TEZ aims to streamline transportation logistics, reduce operational delays, and improve overall user satisfaction by providing real-time information and automating various administrative tasks. By integrating these features into one platform, the TEZ app satisfies corporate goals of improving service quality, ensuring passenger safety, and optimizing resource utilization. It also supports at a large scale business strategies by enhancing operational efficiency, enabling better decision-making, and fostering customer trust through reliable transport management.
The TEZ app fills a significant gap in the local market, where no competing rivals exist, by introducing a system that has been proven effective in international markets.

1.3 Title

TEZ - One stop solution for all public transport needs.

1.4 Objectives

- ***Transportation Management***
  Provide a unified platform for efficiently managing the public transport experience.

- ***Real-Time Bus Tracking***
  Enable passengers to track buses live and receive timely updates on delays or route changes. Help citizens/students organize their time better by knowing their transport vehicle's location and distance, breakdowns, roadblocks, etc.

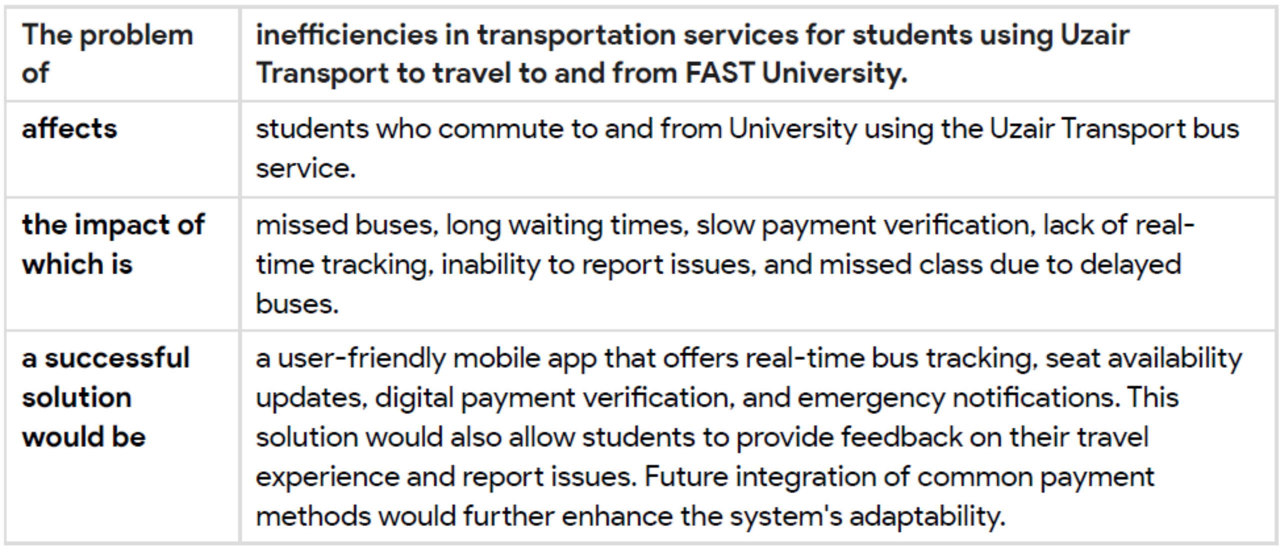
- ***Optimized Resource Use***
  Improve bus utilization by managing seats more optimally and ensuring timely maintenance scheduling.

- ***Enhanced Safety and Convenience***
  Offer features like emergency notifications, secure payments, and digital bus passes for a safer, more convenient experience.

- ***Better Stakeholder Communication***
  Automate key interactions between administrators, drivers, and passengers for smoother operations. This includes Admin assigning buses and routes to drivers, Passengers being able to communicate issues directly with the

- ***Scalable and Adaptable System***
  Design a system that can grow to support a wider range of networks beyond a single organization or a single institute's organizations.

1.5 Problem Statement



The problem of	**inefficiencies in transportation services for students using Uzair Transport to travel to and from FAST University.**
affects	students who commute to and from University using the Uzair Transport bus service.
the impact of which is	missed buses, long waiting times, slow payment verification, lack of real-time tracking, inability to report issues, and missed class due to delayed buses.
a successful solution would be	a user-friendly mobile app that offers real-time bus tracking, seat availability updates, digital payment verification, and emergency notifications. This solution would also allow students to provide feedback on their travel experience and report issues. Future integration of common payment methods would further enhance the system's adaptability.

2. Functional Requirements

The TEZ app must perform or facilitate the following major functions to meet the needs of its users:

3. Nonfunctional Requirements

3.1. Performance Requirements

- ***NFR-P1****: The system must reflect real-time bus location and status updates with a maximum delay of* ***5 minutes****.*
- ***NFR-P2****: Seat availability status must update* ***instantly*** *(within 2 seconds) after reservation or cancellation to avoid overbooking.*
- ***NFR-P3****: System must retrieve user, bus, and schedule data within* ***20 seconds*** *for all admin, driver, and student queries.*

3.2. Usability Requirements

- ***NFR-U1****: The system must have an intuitive UI that requires* ***minimal training****, especially for drivers with limited tech experience.*
- ***NFR-U2****: The user interface should use* ***consistent icons, fonts, and color schemes****, taking inspiration from inDrive for simplicity and familiarity.*
- ***NFR-U3****: All common actions (e.g., "Reserve", "Start Route", "Send Notification") must be executable within* ***3 user interactions (clicks or taps)****.*

3.3. Scalability Requirements

- ***NFR-S1****: The system must handle at least* ***500 concurrent users*** *during peak times (e.g., 7:00–9:00 AM, 4:00–6:00 PM) without performance degradation.*
- ***NFR-S2****: The architecture must support* ***multi-institution scaling****, allowing onboarding of multiple universities or transport providers with isolated data.*

3.4. Security Requirements

- ***NFR-SE1****: All sensitive user data (e.g., personal info, payment details) must be stored using* ***AES-256 encryption*** *or equivalent.*

- ***NFR-SE2****: The system must prevent unauthorized access using* ***role-based access control*** *(RBAC) for students, drivers, and administrators.*
- ***NFR-SE3****: All payment operations must use* ***PCI-DSS-compliant*** *secure payment gateways.*
- ***NFR-SE4****: The app must include* ***fraud detection mechanisms*** *to flag suspicious transactions.*

3.5. Reliability & Availability

- ***NFR-R1****: The app should have* ***99.5% uptime****, excluding scheduled maintenance.*
- ***NFR-R2****: In case of system failure, users must receive a* ***clear error message*** *within 5 seconds along with* ***recovery instructions****.*
- ***NFR-R3****: Feedback and reporting modules must log every submission, even during minor network issues, and* ***retry sending automatically****.*

3.6. Maintainability

- ***NFR-M1****: The system should be modular, enabling updates or bug fixes to one component (e.g., tracking, payments) without affecting others.*
- ***NFR-M2****: Logs and metrics should be kept for* ***debugging and monitoring****, and must be accessible to admins and developers for audits.*

3.7. Interoperability

- ***NFR-I1****: The app must support integration with:*
    - *GPS APIs (e.g., Google Maps)*
    - *Secure Payment Gateways*
    - *University Student Databases (via REST APIs or secure DB connections)*

3.8. Portability

- ***NFR-POR1****: The app must be operable on Android 10+, iOS 13+, and modern desktop browsers (Chrome, Firefox, Safari).*
- ***NFR-POR2****: It must be optimized for both* ***mobile and tablet interfaces****, ensuring full functionality on smaller screens.*

4. User Story

US01: Check Available Seats
User Story: As a student, I want to check available seats in real-time, so that I can decide whether to board the bus.

US02: Track Bus Location
User Story: As a student, I want to track the live location of my bus, so that I can estimate its arrival time.

US03: View Bus Schedule
User Story: As a student, I want to view the schedule of all buses, so that I can plan my commute accordingly.

US04: Select Bus Route
User Story: As a student, I want to see which buses go through my area, so that I can choose the best route.

US05: Receive Emergency Notifications
User Story: As a student, I want to receive alerts about delays or cancellations, so that I can make alternate travel arrangements.

US06: Fingerprint Authentication for Boarding
User Story: As a student, I want to verify my identity using a fingerprint scanner, so that only authorized students can board.

US07: Admin Route Management
User Story: As an admin, I want to set routes and schedules for all buses, so that I can manage transport efficiently.

US08: Admin Notifications
User Story: As an admin, I want to send out notifications to all students and drivers, so that everyone stays informed.

US09: Driver Seat Marking
User Story: As a driver, I want to mark which seats are occupied, so that students know how full the bus is.

US10: Add/Edit/Delete Bus
User Story: As an admin, I want to add, edit, or delete buses, so that I can manage the fleet easily.

US11: Add/Edit/Delete Routes
User Story: As an admin, I want to create and update routes with stops, so that I can accommodate student needs.

US12: View Bus Occupancy
User Story: As a student, I want to view how full a bus is before it arrives, so that I can pick the least crowded one.

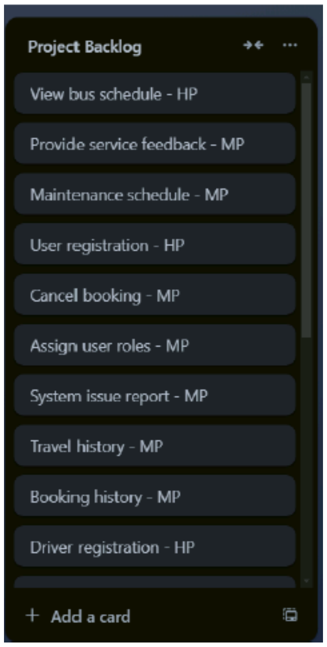
US13: Student Login
User Story: As a student, I want to log into the system securely, so that I can access personalized features.

US14: Admin Login
User Story: As an admin, I want to log into the system securely, so that I can manage system operations.

US15: Driver Login
User Story: As a driver, I want to log into the system securely, so that I can access my assigned routes and seat marking tools.

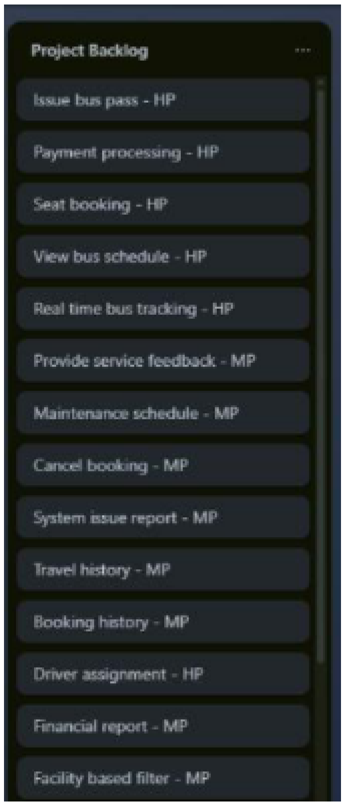
5. Product Backlog

Sprint 1 Backlog


Project Backlog

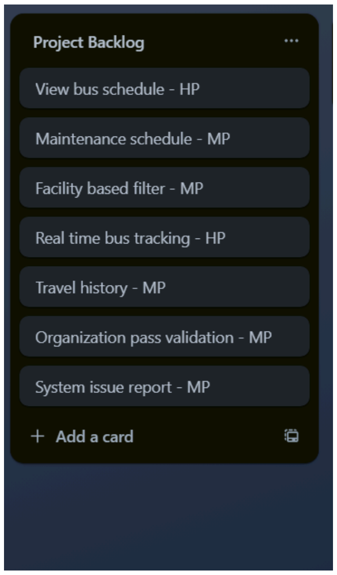
- View bus schedule - HP
- Provide service feedback - MP
- Maintenance schedule - MP
- User registration - HP
- Cancel booking - MP
- Assign user roles - MP
- System issue report - MP
- Travel history - MP
- Booking history - MP
- Driver registration - HP

Sprint 2 Backlog


Project Backlog

- Issue bus pass - HP
- Payment processing - HP
- Seat booking - HP
- View bus schedule - HP
- Real time bus tracking - HP
- Provide service feedback - MP
- Maintenance schedule - MP
- Cancel booking - MP
- System issue report - MP
- Travel history - MP
- Booking history - MP
- Driver assignment - HP
- Financial report - MP
- Facility based filter - MP

Final Backlog



Project Backlog

View bus schedule - HP

Maintenance schedule - MP

Facility based filter - MP

Real time bus tracking - HP

Travel history - MP

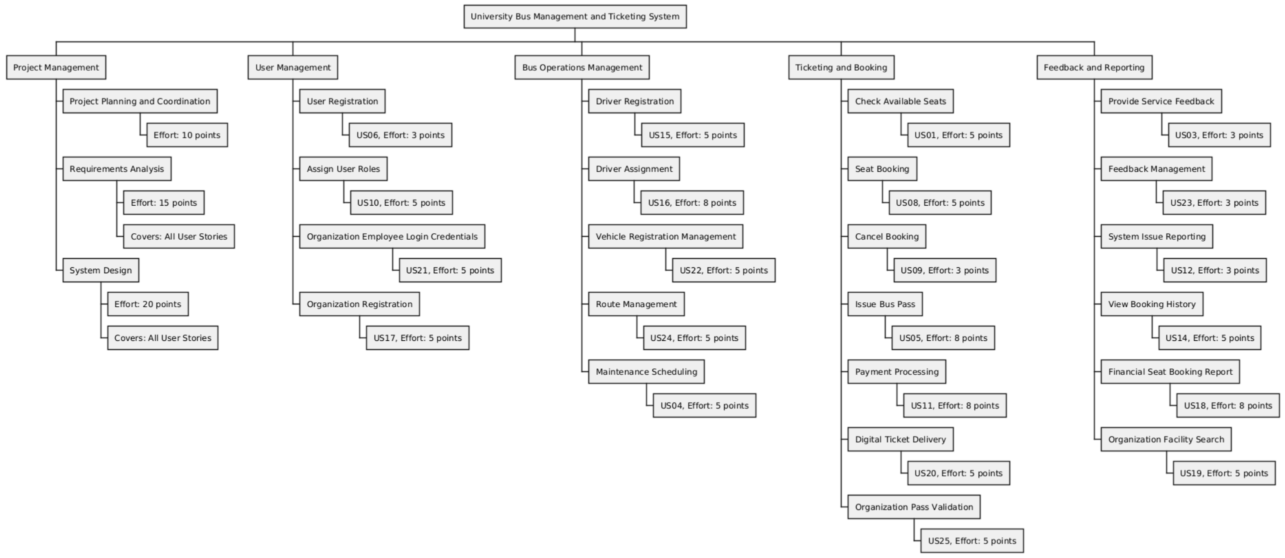
Organization pass validation - MP

System issue report - MP

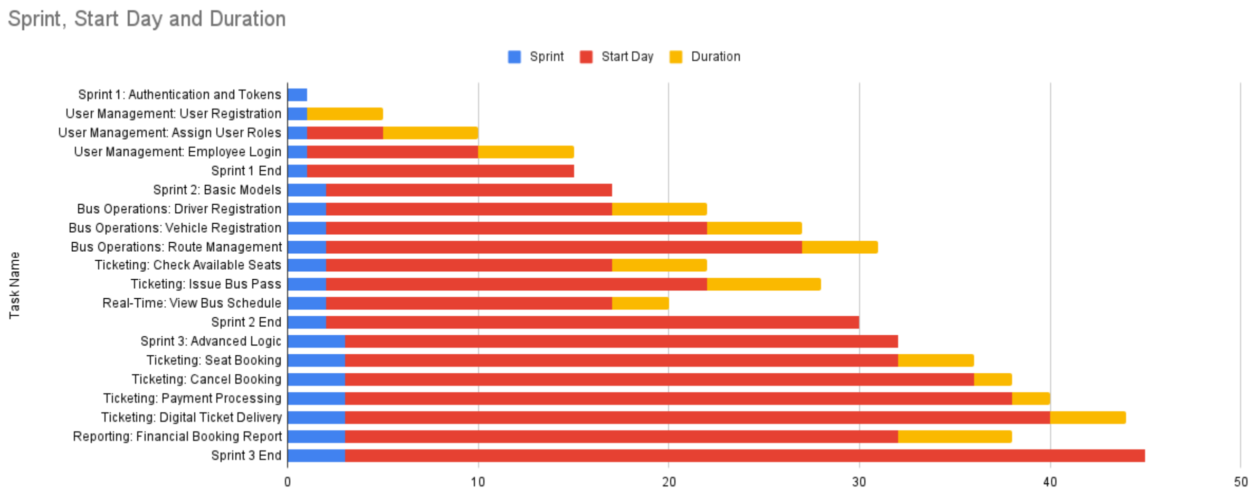
+ Add a card

6. Project Plan

- **WBS Diagram**

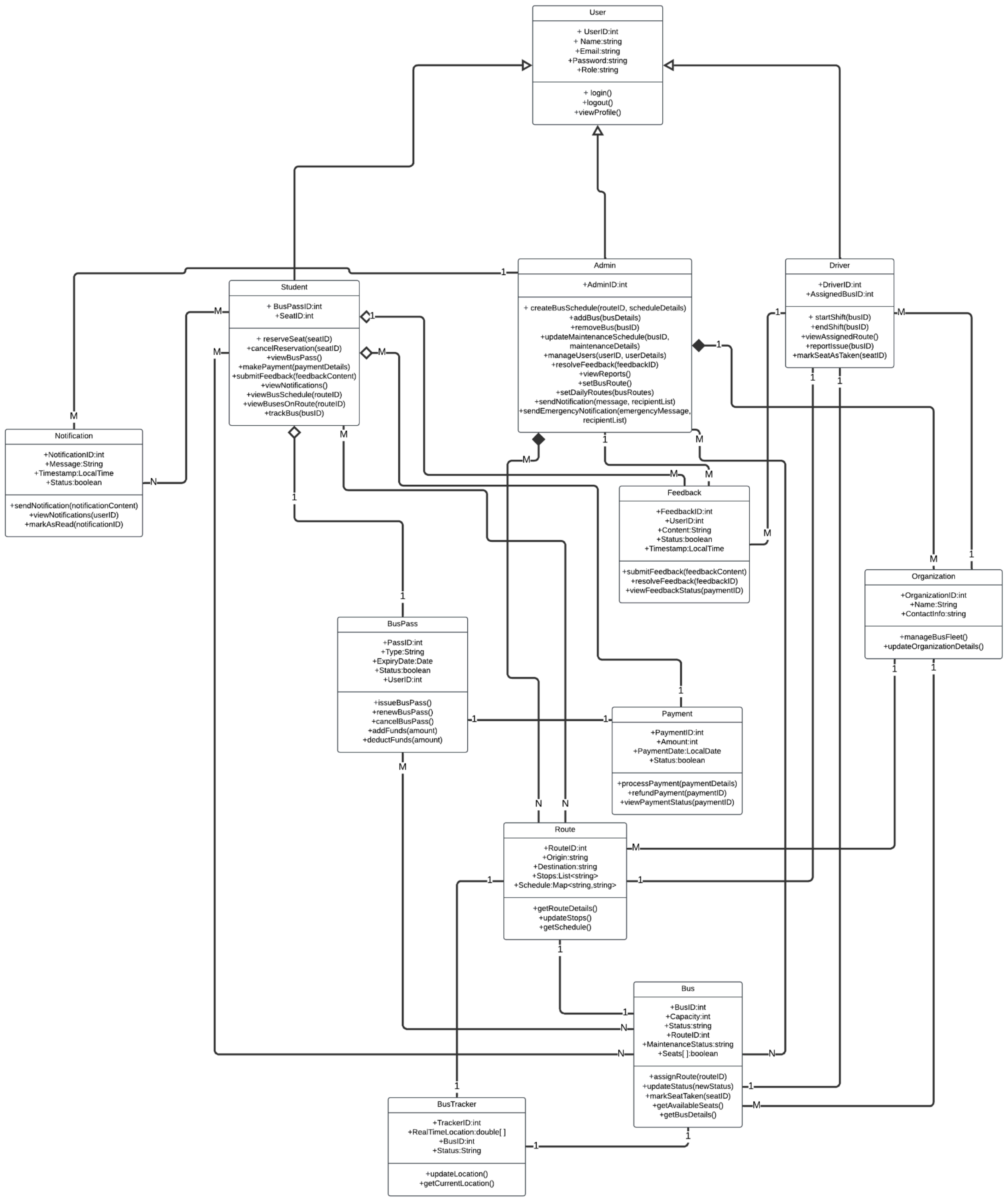


- **Gantt Chart**

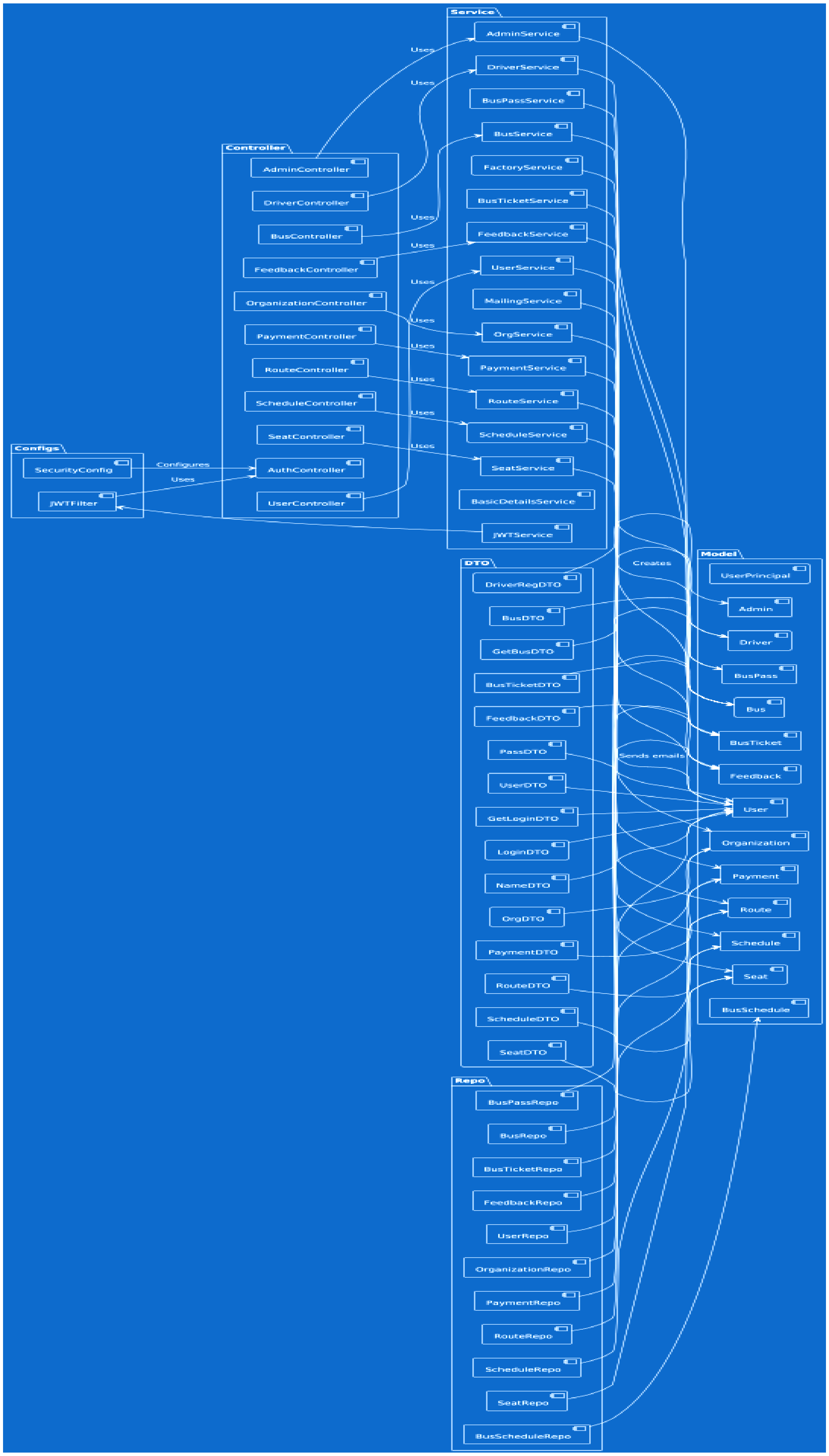


7. Architecture Diagrams

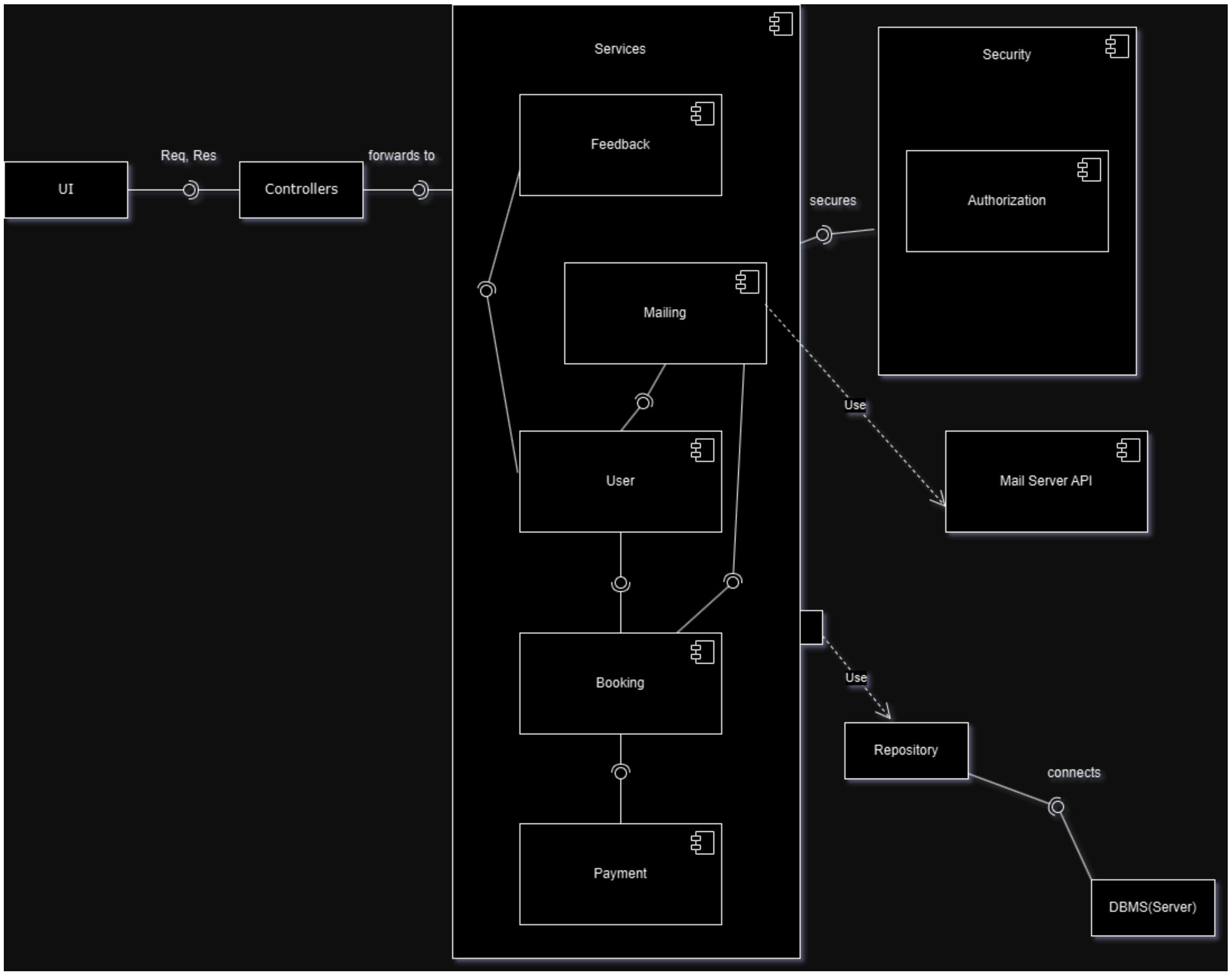
Class Diagram



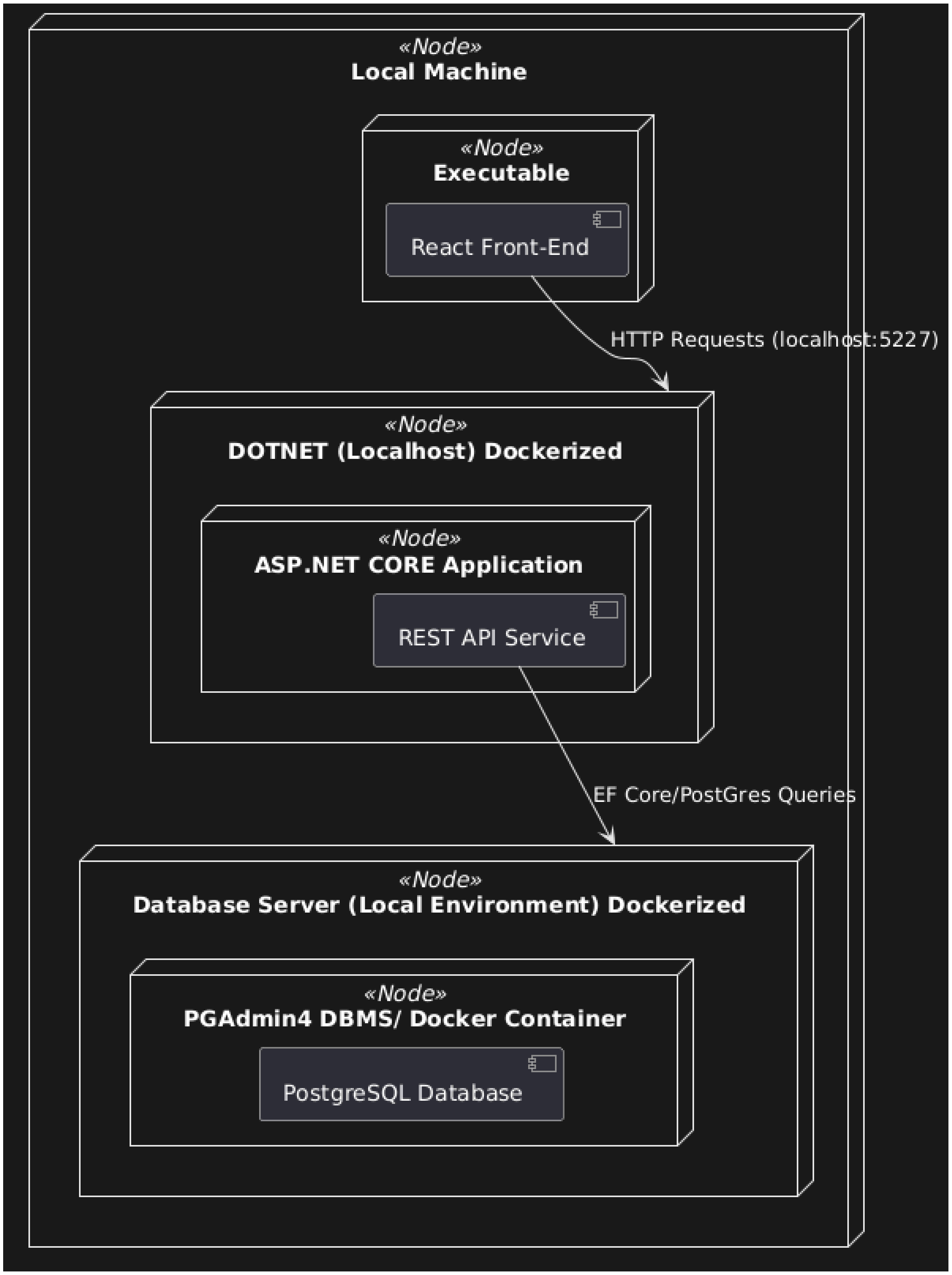
Package Diagram


Service
AdminService
DriverService
BusPassService
BusService
FactoryService
BusTicketService
FeedbackService
UserService
MailingService
OrgService
PaymentService
RouteService
ScheduleService
SeatService
BasicDetailsService
JWTService
Controller
AdminController
DriverController
BusController
FeedbackController
OrganizationController
PaymentController
RouteController
ScheduleController
SeatController
AuthController
UserController
Configs
SecurityConfig
JWTFilter
Configures
Uses
DTO
DriverRegDTO
BusDTO
GetBusDTO
BusTicketDTO
FeedbackDTO
PassDTO
UserDTO
GetLoginDTO
LoginDTO
NameDTO
OrgDTO
PaymentDTO
RouteDTO
ScheduleDTO
SeatDTO
Creates
Sends emails
Model
UserPrincipal
Admin
Driver
BusPass
Bus
BusTicket
Feedback
User
Organization
Payment
Route
Schedule
Seat
BusSchedule
Repo
BusPassRepo
BusRepo
BusTicketRepo
FeedbackRepo
UserRepo
OrganizationRepo
PaymentRepo
RouteRepo
ScheduleRepo
SeatRepo
BusScheduleRepo

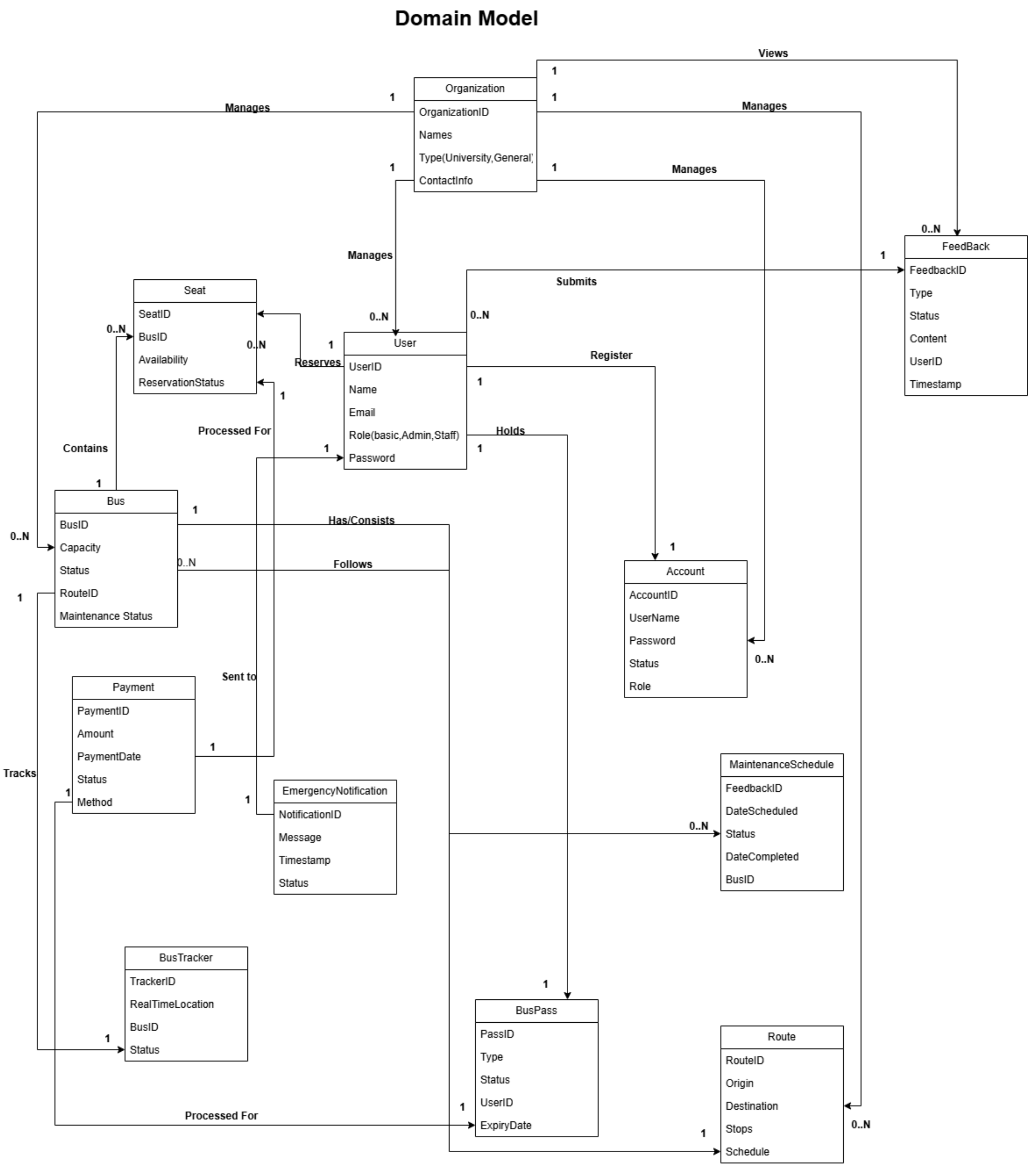
Component Diagram



Deployment Diagram

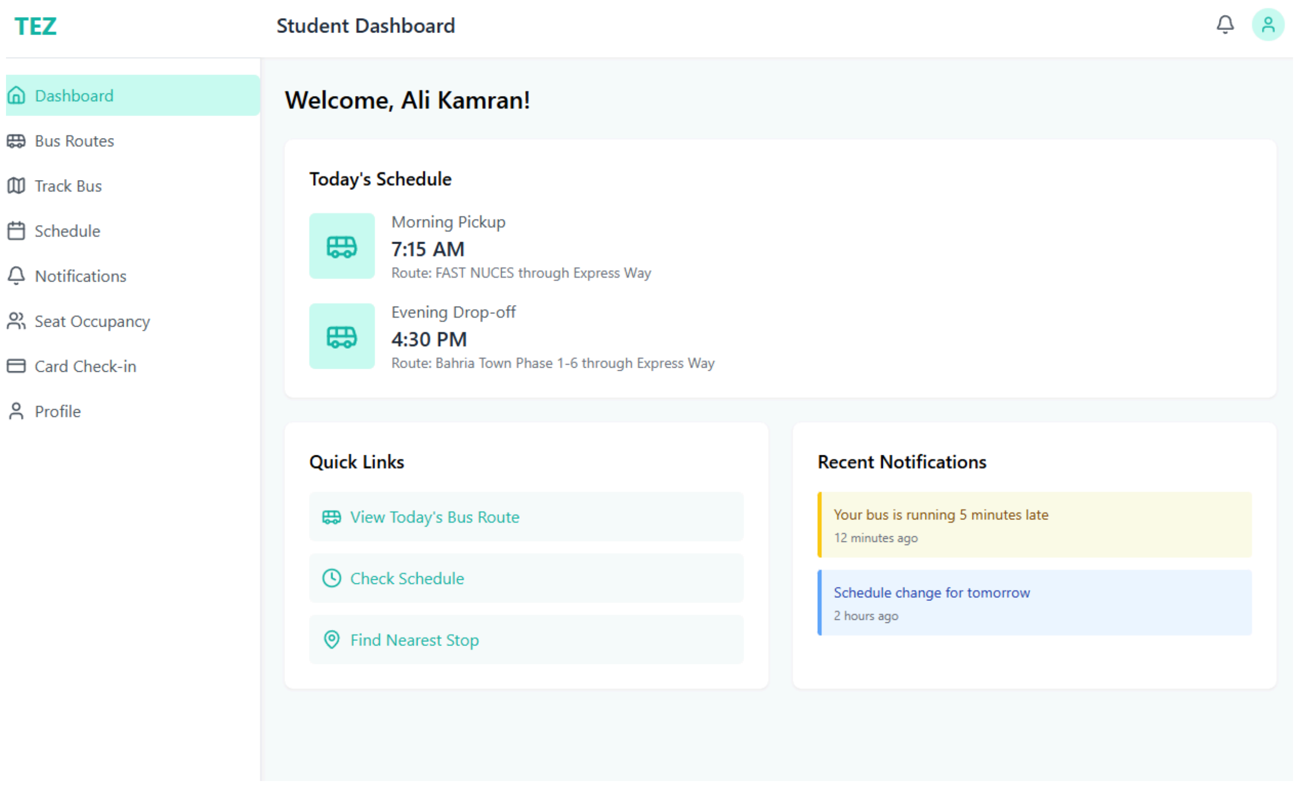

«Node»
Local Machine
«Node»
Executable
React Front-End
HTTP Requests (localhost:5227)
«Node»
DOTNET (Localhost) Dockerized
«Node»
ASP.NET CORE Application
REST API Service
EF Core/PostGres Queries
«Node»
Database Server (Local Environment) Dockerized
«Node»
PGAdmin4 DBMS/ Docker Container
PostgreSQL Database

Domain Model



8. Design and Implementation Screenshots

FrontEnd


TEZ

Student Dashboard

Dashboard
Bus Routes
Track Bus
Schedule
Notifications
Seat Occupancy
Card Check-in
Profile

Welcome, Ali Kamran!

Today's Schedule

Morning Pickup
7:15 AM
Route: FAST NUCES through Express Way

Evening Drop-off
4:30 PM
Route: Bahria Town Phase 1-6 through Express Way

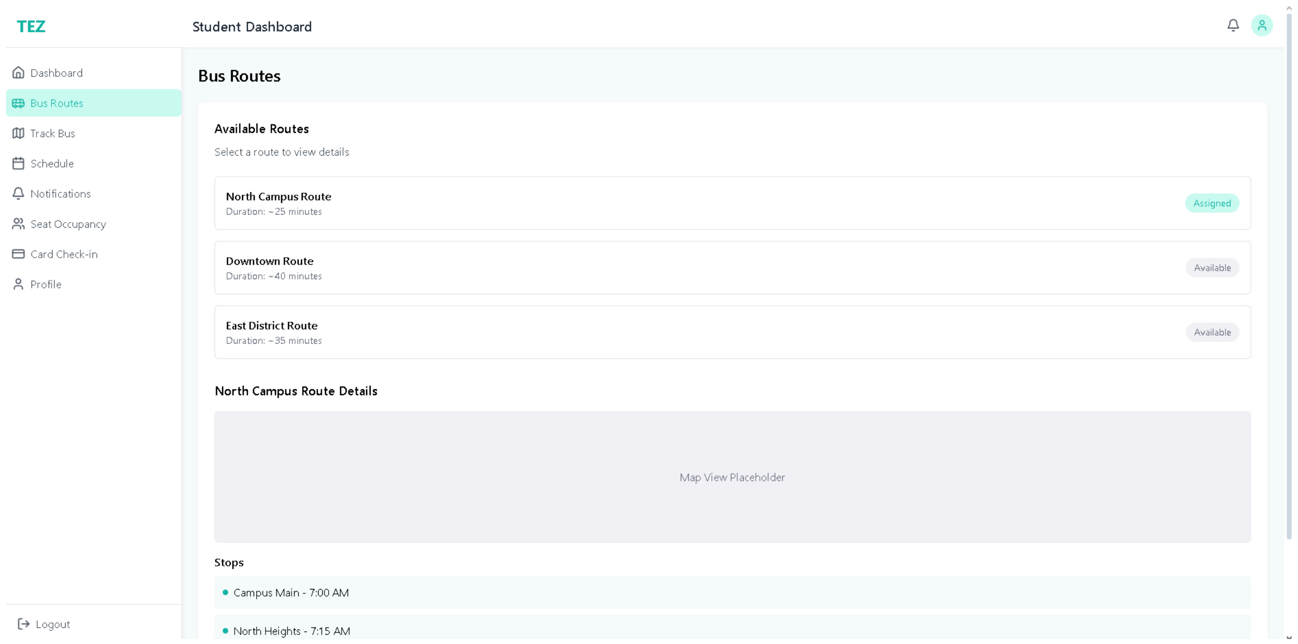
Quick Links

View Today's Bus Route
Check Schedule
Find Nearest Stop

Recent Notifications

Your bus is running 5 minutes late
12 minutes ago

Schedule change for tomorrow
2 hours ago


TEZ

Student Dashboard

Dashboard
Bus Routes
Track Bus
Schedule
Notifications
Seat Occupancy
Card Check-in
Profile

Logout

Bus Routes

Available Routes
Select a route to view details

North Campus Route
Duration: ~25 minutes
Assigned

Downtown Route
Duration: ~40 minutes
Available

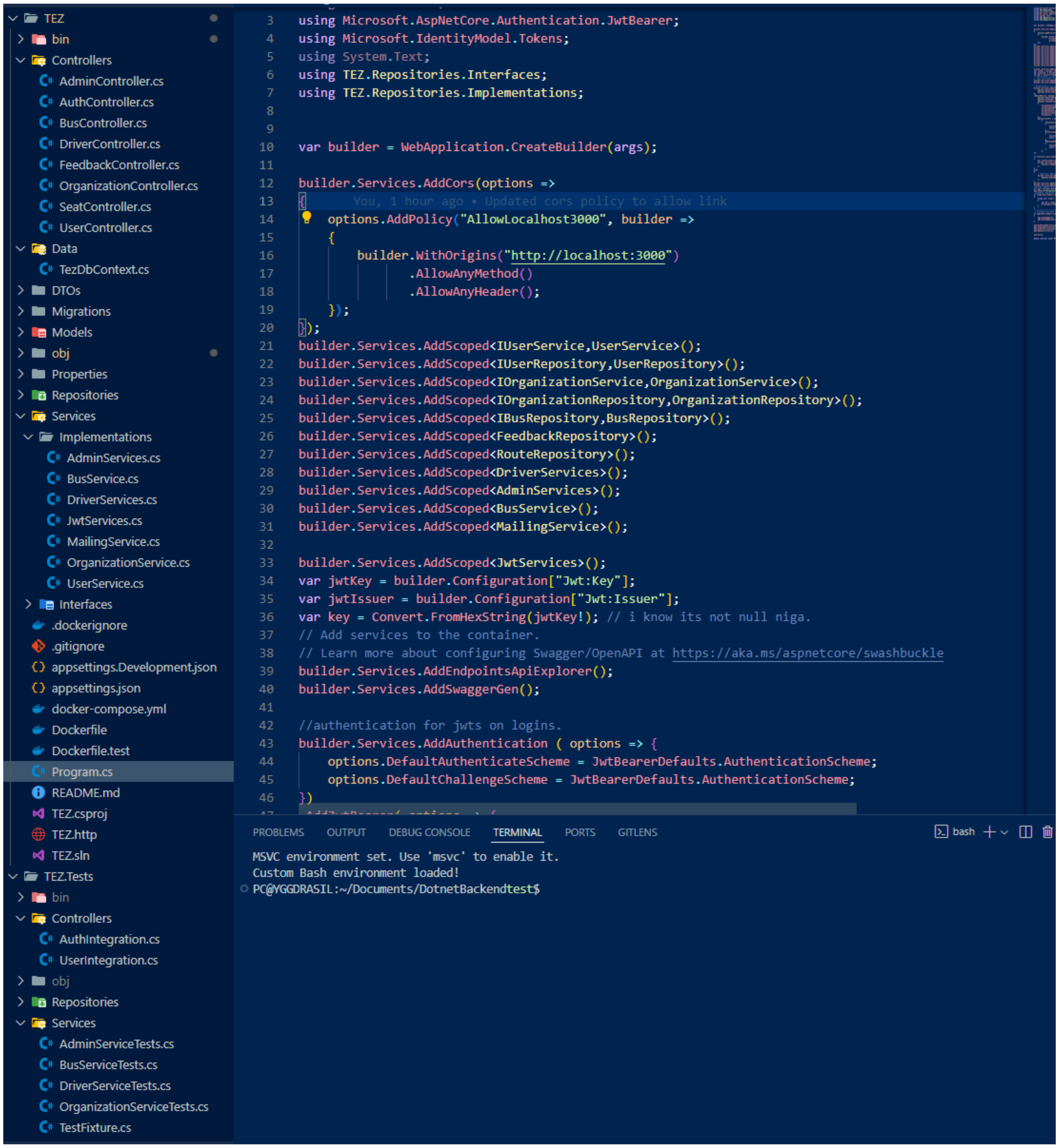
East District Route
Duration: ~35 minutes
Available

North Campus Route Details

Map View Placeholder

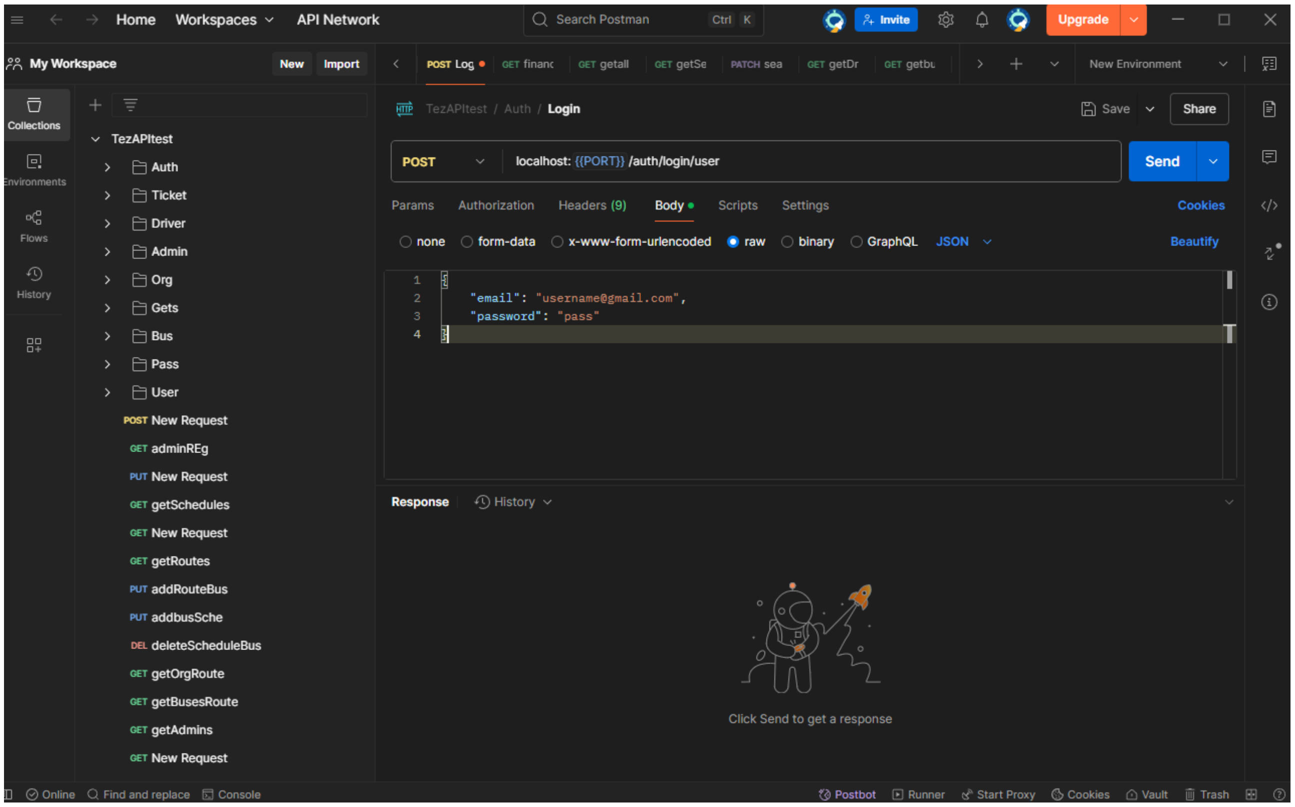
Stops
- Campus Main - 7:00 AM
- North Heights - 7:15 AM

Backend



```
using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.IdentityModel.Tokens;
using System.Text;
using TEZ.Repositories.Interfaces;
using TEZ.Repositories.Implementations;


var builder = WebApplication.CreateBuilder(args);

builder.Services.AddCors(options =>
{
    options.AddPolicy("AllowLocalhost3000", builder =>
    {
        builder.WithOrigins("http://localhost:3000")
                .AllowAnyMethod()
                .AllowAnyHeader();
    });
});
builder.Services.AddScoped<IUserService,UserService>();
builder.Services.AddScoped<IUserRepository,UserRepository>();
builder.Services.AddScoped<IOrganizationService,OrganizationService>();
builder.Services.AddScoped<IOrganizationRepository,OrganizationRepository>();
builder.Services.AddScoped<IBusRepository,BusRepository>();
builder.Services.AddScoped<FeedbackRepository>();
builder.Services.AddScoped<RouteRepository>();
builder.Services.AddScoped<DriverServices>();
builder.Services.AddScoped<AdminServices>();
builder.Services.AddScoped<BusService>();
builder.Services.AddScoped<MailingService>();

builder.Services.AddScoped<JwtServices>();
var jwtKey = builder.Configuration["Jwt:Key"];
var jwtIssuer = builder.Configuration["Jwt:Issuer"];
var key = Convert.FromHexString(jwtKey!); // i know its not null niga.
// Add services to the container.
// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();

//authentication for jwts on logins.
builder.Services.AddAuthentication ( options => {
    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
    options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
})
```

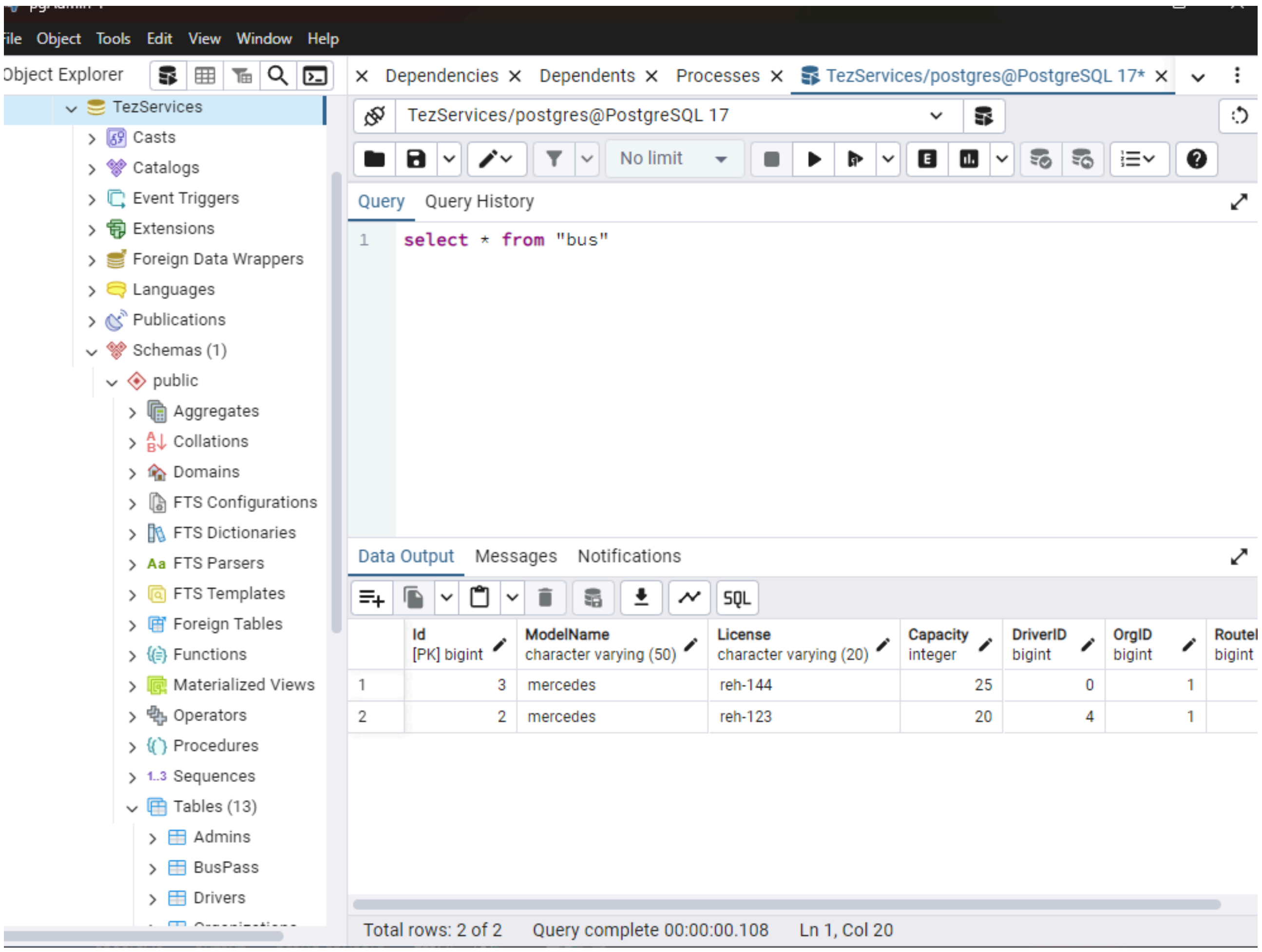
```
MSVC environment set. Use 'msvc' to enable it.
Custom Bash environment loaded!
PC@YGGDRASIL:~/Documents/DotnetBackendtest$
```



POST localhost: {{PORT}} /auth/login/user

```
{
    "email": "username@gmail.com",
    "password": "pass"
}
```

DBMS

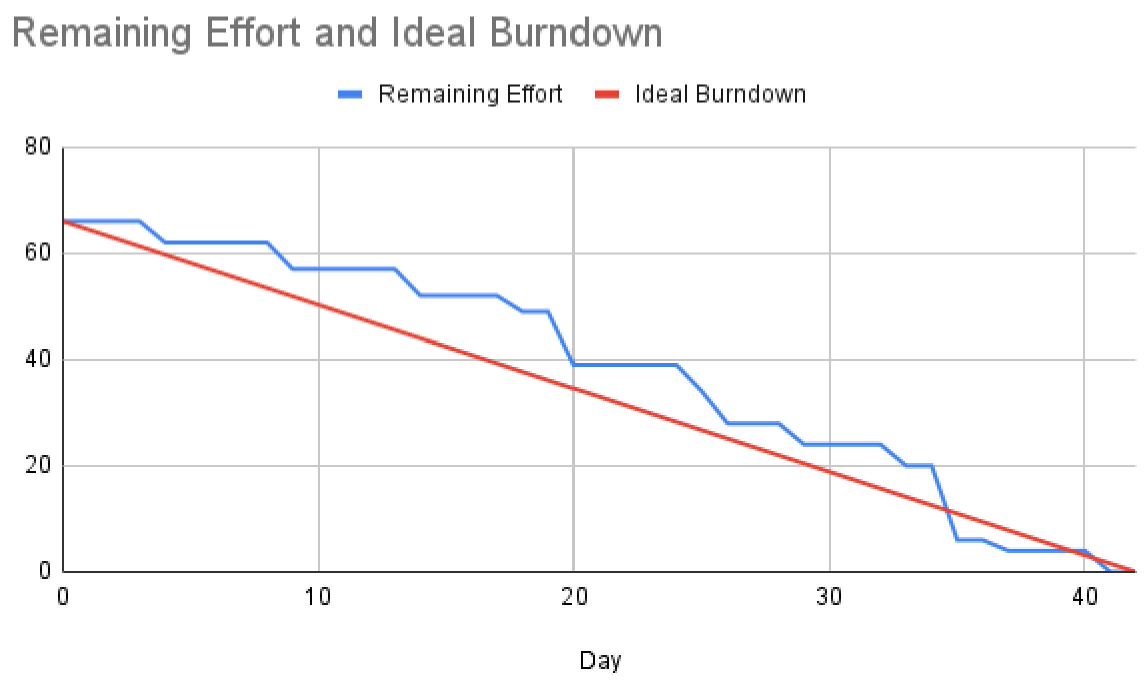


```
select * from "bus"
```

	Id [PK] bigint	ModelName character varying (50)	License character varying (20)	Capacity integer	DriverID bigint	OrgID bigint	RouteI bigint
1	3	mercedes	reh-144	25	0	1	
2	2	mercedes	reh-123	20	4	1	

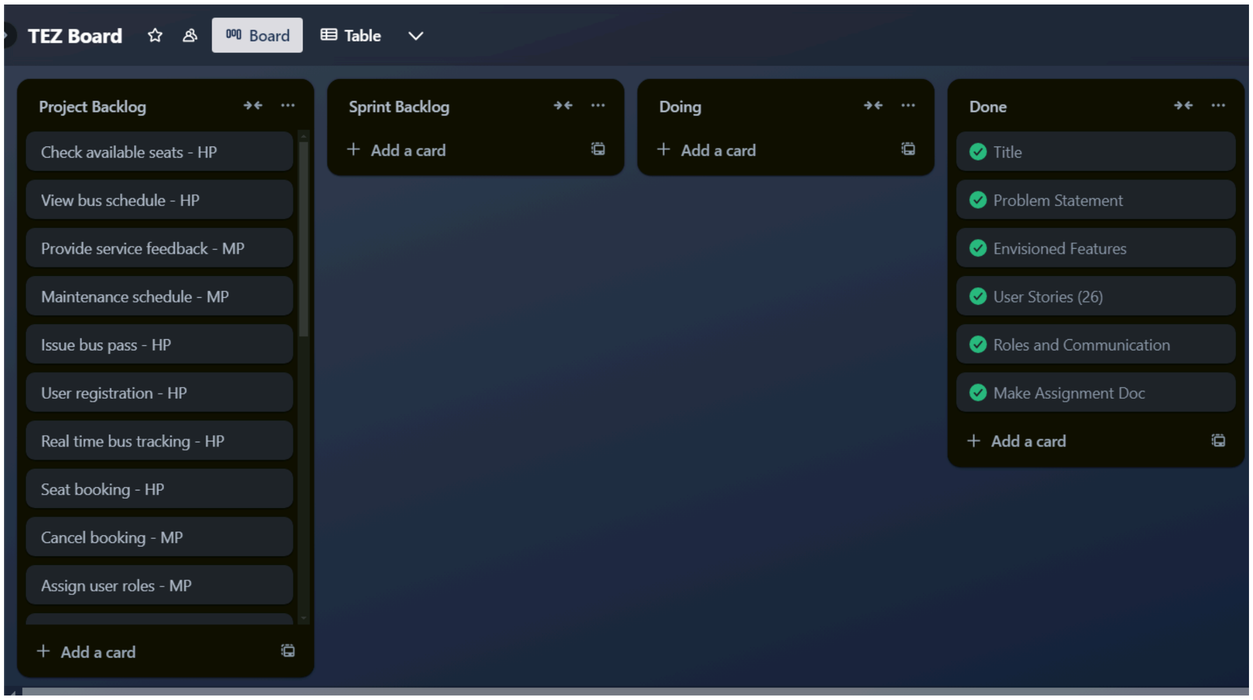
Total rows: 2 of 2 Query complete 00:00:00.108 Ln 1, Col 20

9. Product Burndown chart for the project



While the overall trend follows the expected decline, the actual effort line shows periods of stagnation—particularly in the first half—indicating delays or blocked tasks. However, the team managed to recover in the latter stages with more consistent progress, ultimately meeting the target by the end. This reflects our improved sprint planning, better task execution in the second half, and adaptability under the Scrum framework, even when faced with initial setbacks.

10. Trello board screenshots

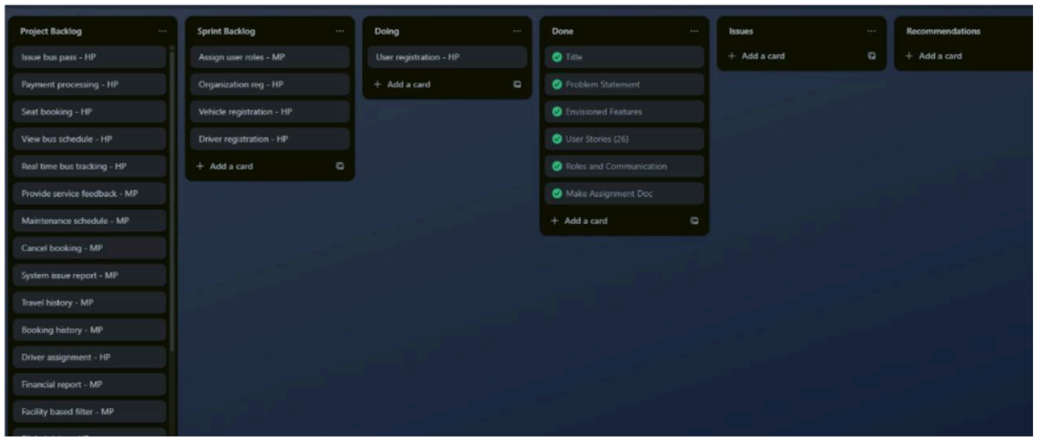


Project Backlog
- Check available seats - HP
- View bus schedule - HP
- Provide service feedback - MP
- Maintenance schedule - MP
- Issue bus pass - HP
- User registration - HP
- Real time bus tracking - HP
- Seat booking - HP
- Cancel booking - MP
- Assign user roles - MP
- + Add a card

Sprint Backlog
- + Add a card

Doing
- + Add a card

Done
- ✓ Title
- ✓ Problem Statement
- ✓ Envisioned Features
- ✓ User Stories (26)
- ✓ Roles and Communication
- ✓ Make Assignment Doc
- + Add a card



Project Backlog
- Issue bus pass - HP
- Payment processing - HP
- Seat booking - HP
- View bus schedule - HP
- Real time bus tracking - HP
- Provide service feedback - MP
- Maintenance schedule - MP
- Cancel booking - MP
- System issue report - MP
- Travel history - MP
- Booking history - MP
- Driver assignment - HP
- Financial report - MP
- Facility based filter - MP

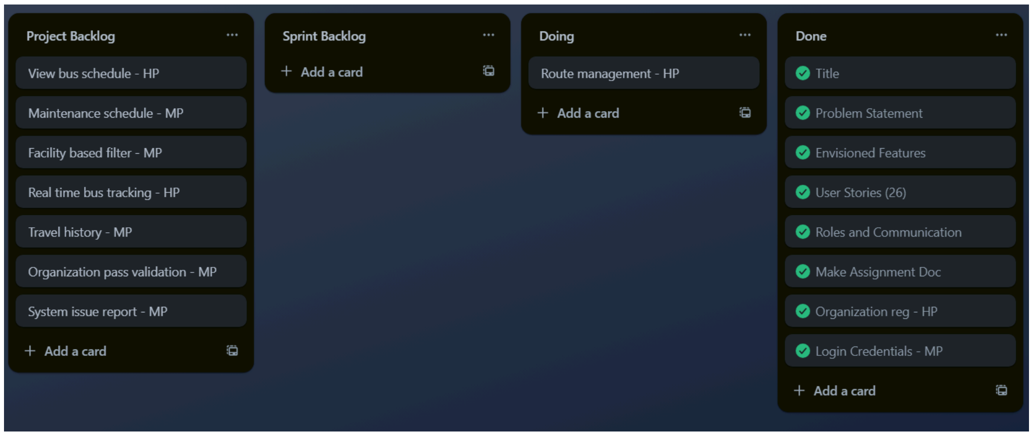
Sprint Backlog
- Assign user roles - MP
- Organization reg - HP
- Vehicle registration - HP
- Driver registration - HP
- + Add a card

Doing
- User registration - HP
- + Add a card

Done
- ✓ Title
- ✓ Problem Statement
- ✓ Envisioned Features
- ✓ User Stories (26)
- ✓ Roles and Communication
- ✓ Make Assignment Doc
- + Add a card

Issues
- + Add a card

Recommendations
- + Add a card

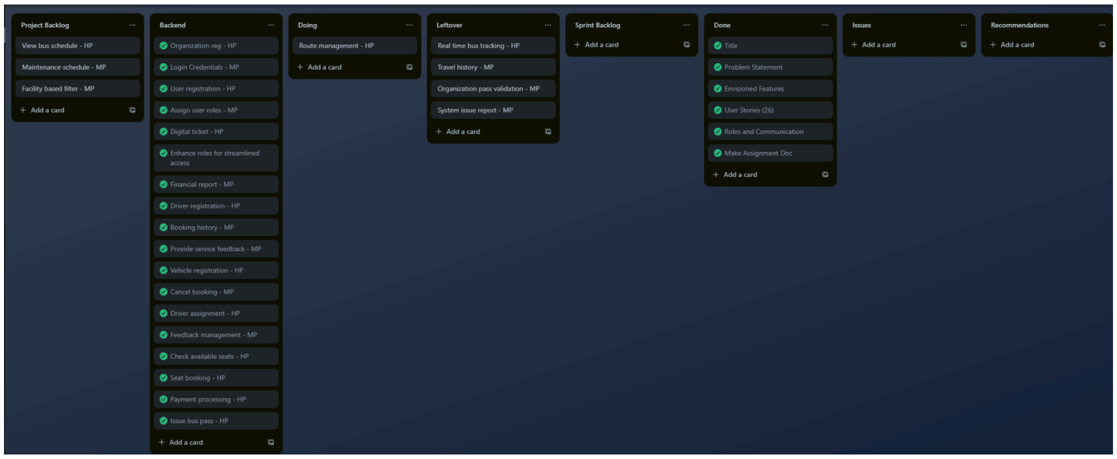


Project Backlog
- View bus schedule - HP
- Maintenance schedule - MP
- Facility based filter - MP
- Real time bus tracking - HP
- Travel history - MP
- Organization pass validation - MP
- System issue report - MP
- + Add a card

Sprint Backlog
- + Add a card

Doing
- Route management - HP
- + Add a card

Done
- ✓ Title
- ✓ Problem Statement
- ✓ Envisioned Features
- ✓ User Stories (26)
- ✓ Roles and Communication
- ✓ Make Assignment Doc
- ✓ Organization reg - HP
- ✓ Login Credentials - MP
- + Add a card



Project Backlog
- View bus schedule - HP
- Maintenance schedule - MP
- Facility based filter - MP

Backend
- ✓ Organization reg - HP
- ✓ Login Credentials - MP
- ✓ User registration - HP
- ✓ Assign user roles - MP
- ✓ Digital ticket - HP
- ✓ Enhance roles for streamlined access
- ✓ Financial report - MP
- ✓ Driver registration - HP
- ✓ Booking history - MP
- ✓ Provide service feedback - MP
- ✓ Vehicle registration - HP
- ✓ Cancel booking - MP
- ✓ Driver assignment - HP
- ✓ Feedback management - MP
- ✓ Check available seats - HP
- ✓ Seat booking - HP
- ✓ Payment processing - HP
- ✓ Issue bus pass - HP

Doing
- Route management - HP

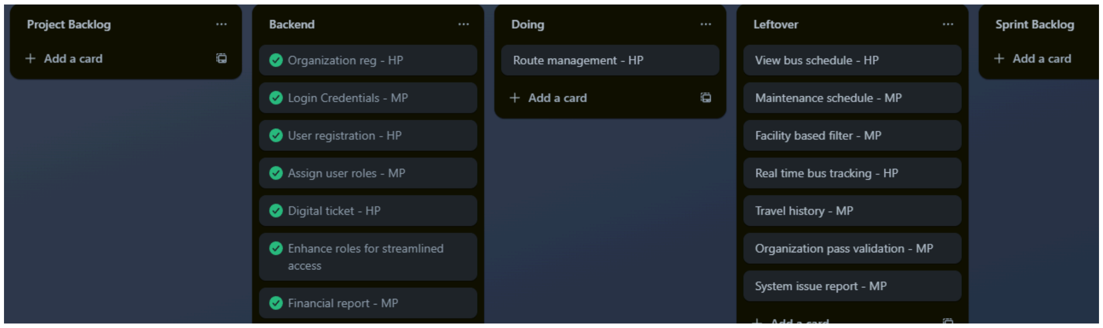
Leftover
- Real time bus tracking - HP
- Travel history - MP
- Organization pass validation - MP
- System issue report - MP

Sprint Backlog

Done
- ✓ Title
- ✓ Problem Statement
- ✓ Envisioned Features
- ✓ User Stories (26)
- ✓ Roles and Communication
- ✓ Make Assignment Doc

Issues

Recommendations



Project Backlog

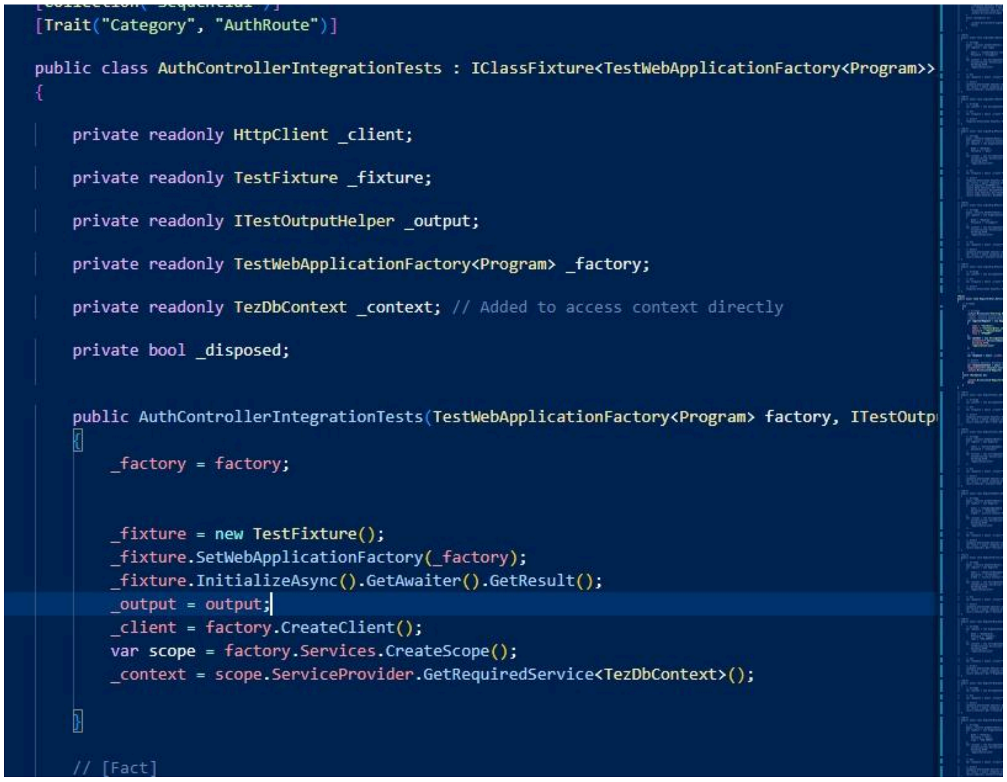
Backend
- ✓ Organization reg - HP
- ✓ Login Credentials - MP
- ✓ User registration - HP
- ✓ Assign user roles - MP
- ✓ Digital ticket - HP
- ✓ Enhance roles for streamlined access
- ✓ Financial report - MP

Doing
- Route management - HP

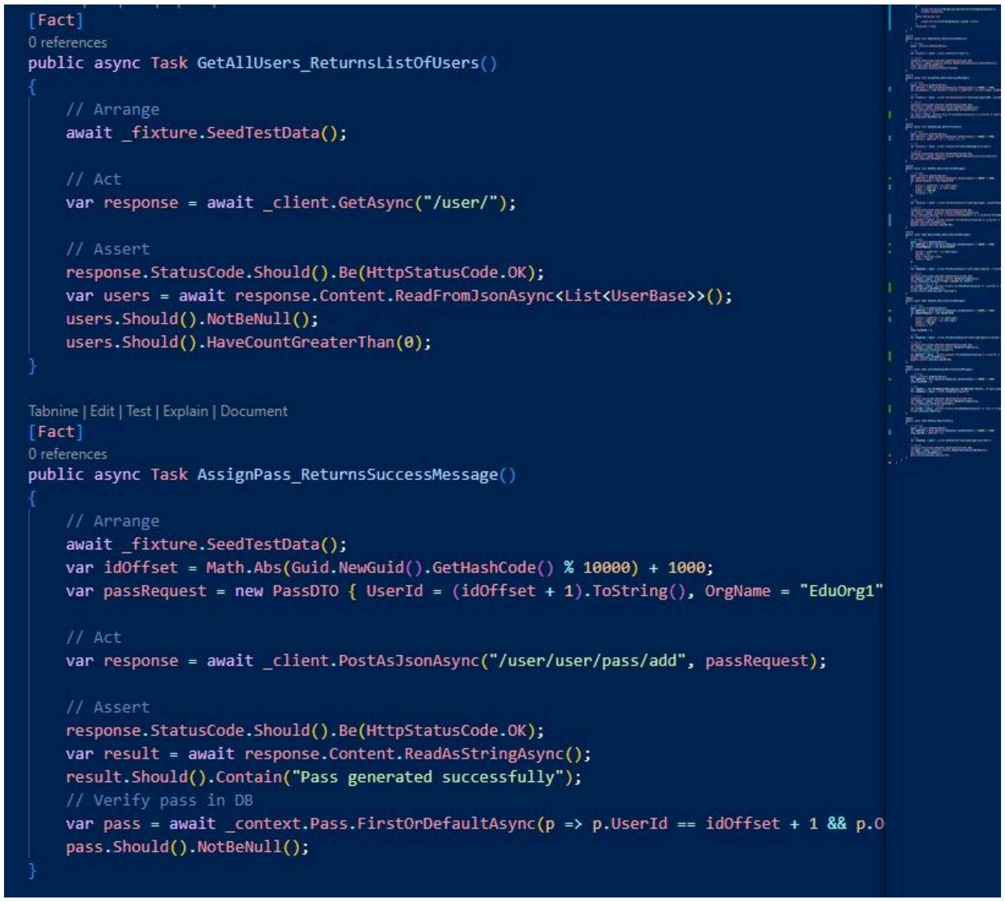
Leftover
- View bus schedule - HP
- Maintenance schedule - MP
- Facility based filter - MP
- Real time bus tracking - HP
- Travel history - MP
- Organization pass validation - MP
- System issue report - MP

Sprint Backlog

11. Testcases -Black box

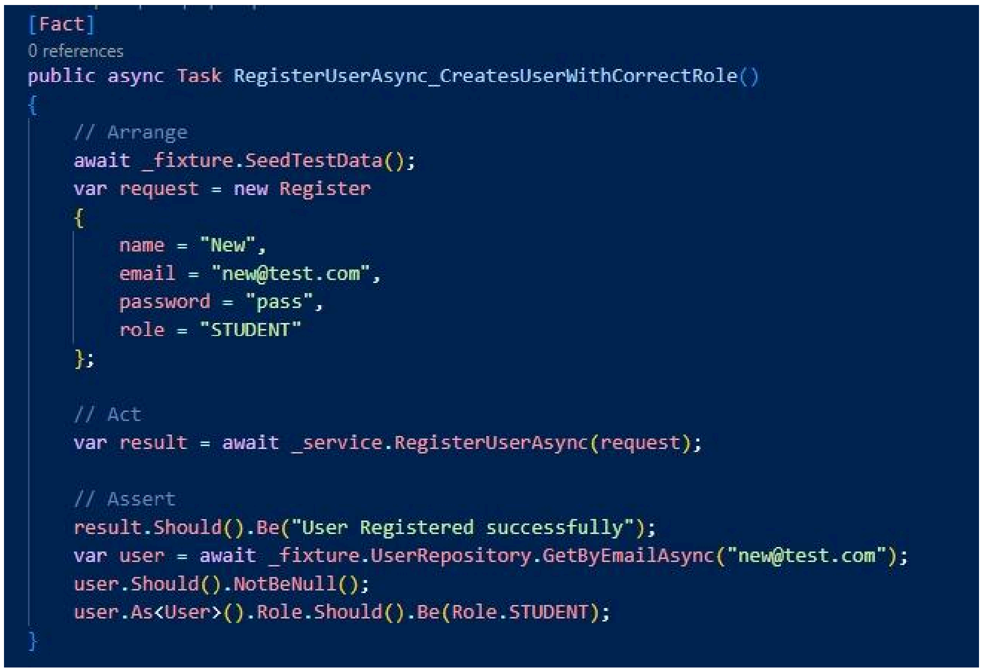


```
[Trait("Category", "AuthRoute")]

public class AuthControllerIntegrationTests : IClassFixture<TestWebApplicationFactory<Program>>
{

    private readonly HttpClient _client;

    private readonly TestFixture _fixture;

    private readonly ITestOutputHelper _output;

    private readonly TestWebApplicationFactory<Program> _factory;

    private readonly TezDbContext _context; // Added to access context directly

    private bool _disposed;


    public AuthControllerIntegrationTests(TestWebApplicationFactory<Program> factory, ITestOutp
    {
        _factory = factory;


        _fixture = new TestFixture();
        _fixture.SetWebApplicationFactory(_factory);
        _fixture.InitializeAsync().GetAwaiter().GetResult();
        _output = output;
        _client = factory.CreateClient();
        var scope = factory.Services.CreateScope();
        _context = scope.ServiceProvider.GetRequiredService<TezDbContext>();

    }

    // [Fact]
```

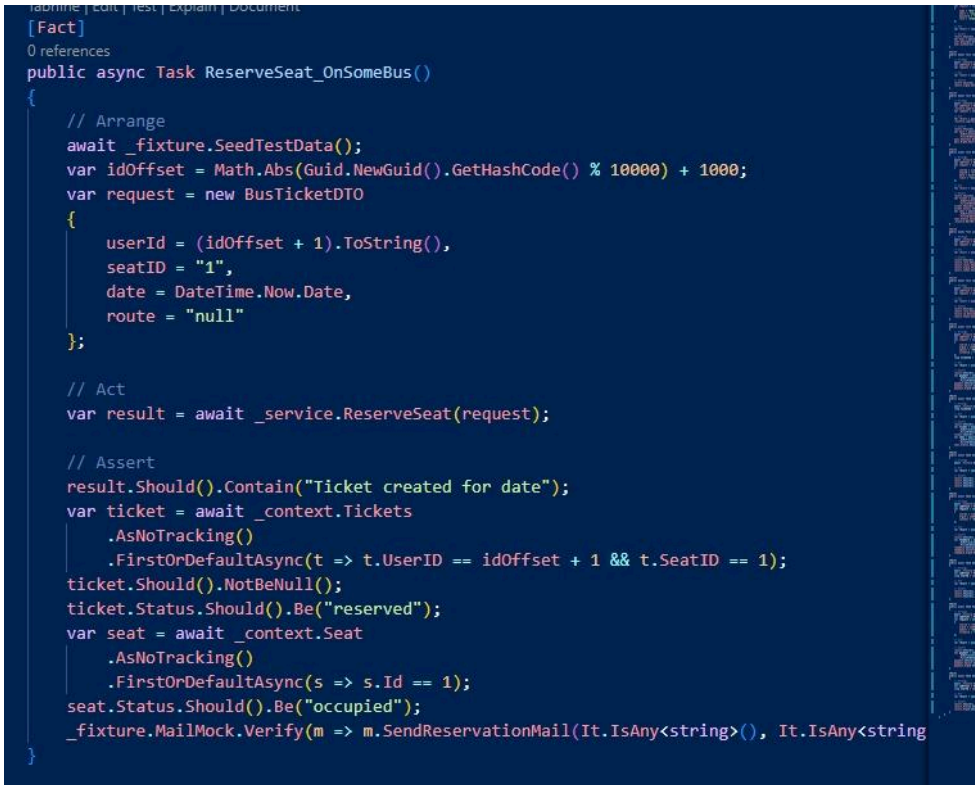


```
[Fact]
0 references
public async Task GetAllUsers_ReturnsListOfUsers()
{
    // Arrange
    await _fixture.SeedTestData();

    // Act
    var response = await _client.GetAsync("/user/");

    // Assert
    response.StatusCode.Should().Be(HttpStatusCode.OK);
    var users = await response.Content.ReadFromJsonAsync<List<UserBase>>();
    users.Should().NotBeNull();
    users.Should().HaveCountGreaterThan(0);
}

Tabnine | Edit | Test | Explain | Document
[Fact]
0 references
public async Task AssignPass_ReturnsSuccessMessage()
{
    // Arrange
    await _fixture.SeedTestData();
    var idOffset = Math.Abs(Guid.NewGuid().GetHashCode() % 10000) + 1000;
    var passRequest = new PassDTO { UserId = (idOffset + 1).ToString(), OrgName = "EduOrg1"

    // Act
    var response = await _client.PostAsJsonAsync("/user/user/pass/add", passRequest);

    // Assert
    response.StatusCode.Should().Be(HttpStatusCode.OK);
    var result = await response.Content.ReadAsStringAsync();
    result.Should().Contain("Pass generated successfully");
    // Verify pass in DB
    var pass = await _context.Pass.FirstOrDefaultAsync(p => p.UserId == idOffset + 1 && p.O
    pass.Should().NotBeNull();
}
```

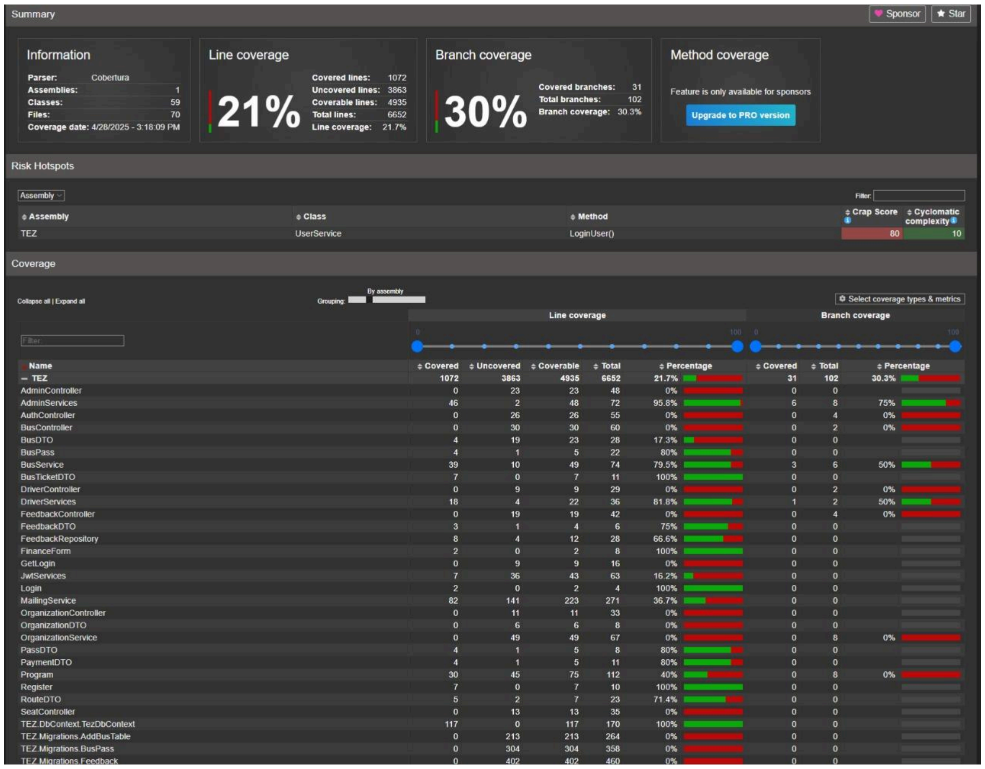
12. Testcases -White box



```
[Fact]
0 references
public async Task RegisterUserAsync_CreatesUserWithCorrectRole()
{
    // Arrange
    await _fixture.SeedTestData();
    var request = new Register
    {
        name = "New",
        email = "new@test.com",
        password = "pass",
        role = "STUDENT"
    };

    // Act
    var result = await _service.RegisterUserAsync(request);

    // Assert
    result.Should().Be("User Registered successfully");
    var user = await _fixture.UserRepository.GetByEmailAsync("new@test.com");
    user.Should().NotBeNull();
    user.As<User>().Role.Should().Be(Role.STUDENT);
}
```



```
Tabnine | Edit | Test | Explain | Document
[Fact]
0 references
public async Task ReserveSeat_OnSomeBus()
{
    // Arrange
    await _fixture.SeedTestData();
    var idOffset = Math.Abs(Guid.NewGuid().GetHashCode() % 10000) + 1000;
    var request = new BusTicketDTO
    {
        userId = (idOffset + 1).ToString(),
        seatID = "1",
        date = DateTime.Now.Date,
        route = "null"
    };

    // Act
    var result = await _service.ReserveSeat(request);

    // Assert
    result.Should().Contain("Ticket created for date");
    var ticket = await _context.Tickets
        .AsNoTracking()
        .FirstOrDefaultAsync(t => t.UserID == idOffset + 1 && t.SeatID == 1);
    ticket.Should().NotBeNull();
    ticket.Status.Should().Be("reserved");
    var seat = await _context.Seat
        .AsNoTracking()
        .FirstOrDefaultAsync(s => s.Id == 1);
    seat.Status.Should().Be("occupied");
    _fixture.MailMock.Verify(m => m.SendReservationMail(It.IsAny<string>(), It.IsAny<string
}
```



Summary

Sponsor | Star

Information

Parser:	Cobertura
Assemblies:	1
Classes:	59
Files:	70
Coverage date:	4/28/2025 - 3:18:09 PM

Line coverage

21%

Covered lines:	1072
Uncovered lines:	3863
Coverable lines:	4935
Total lines:	6652
Line coverage:	21.7%

Branch coverage

30%

Covered branches:	31
Total branches:	102
Branch coverage:	30.3%

Method coverage

Feature is only available for sponsors

Upgrade to PRO version

Risk Hotspots

Assembly Filter:

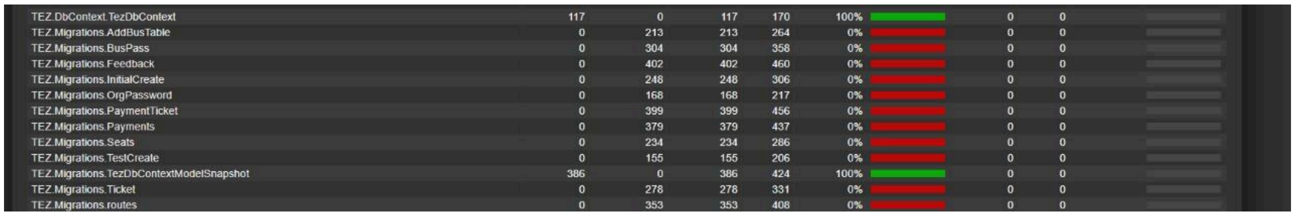
Assembly	Class	Method	Crap Score	Cyclomatic complexity
TEZ	UserService	LoginUser()	80	10

Coverage

Collapse all \| Expand all Grouping: By assembly Select coverage types & metrics

Filter

	Line coverage					Branch coverage		
Name	Covered	Uncovered	Coverable	Total	Percentage	Covered	Total	Percentage
— TEZ	1072	3863	4935	6652	21.7%	31	102	30.3%
AdminController	0	23	23	48	0%	0	0	
AdminServices	46	2	48	72	95.8%	6	8	75%
AuthController	0	26	26	55	0%	0	4	0%
BusController	0	30	30	60	0%	0	2	0%
BusDTO	4	19	23	28	17.3%	0	0	
BusPass	4	1	5	22	80%	0	0	
BusService	39	10	49	74	79.5%	3	6	50%
BusTicketDTO	7	0	7	11	100%	0	0	
DriverController	0	9	9	29	0%	0	2	0%
DriverServices	18	4	22	36	81.8%	1	2	50%
FeedbackController	0	19	19	42	0%	0	4	0%
FeedbackDTO	3	1	4	6	75%	0	0	
FeedbackRepository	8	4	12	28	66.6%	0	0	
FinanceForm	2	0	2	8	100%	0	0	
GetLogin	0	9	9	16	0%	0	0	
JwtServices	7	36	43	63	16.2%	0	0	
Login	2	0	2	4	100%	0	0	
MailingService	82	141	223	271	36.7%	0	0	
OrganizationController	0	11	11	33	0%	0	0	
OrganizationDTO	0	6	6	8	0%	0	0	
OrganizationService	0	49	49	67	0%	0	8	0%
PassDTO	4	1	5	8	80%	0	0	
PaymentDTO	4	1	5	11	80%	0	0	
Program	30	45	75	112	40%	0	8	0%
Register	7	0	7	10	100%	0	0	
RouteDTO	5	2	7	23	71.4%	0	0	
SeatController	0	13	13	35	0%	0	0	
TEZ.DbContext.TezDbContext	117	0	117	170	100%	0	0	
TEZ.Migrations.AddBusTable	0	213	213	264	0%	0	0	
TEZ.Migrations.BusPass	0	304	304	358	0%	0	0	
TEZ.Migrations.Feedback	0	402	402	460	0%	0	0	



TEZ.DbContext.TezDbContext	117	0	117	170	100%	0	0	
TEZ.Migrations.AddBusTable	0	213	213	264	0%	0	0	
TEZ.Migrations.BusPass	0	304	304	358	0%	0	0	
TEZ.Migrations.Feedback	0	402	402	460	0%	0	0	
TEZ.Migrations.InitialCreate	0	248	248	306	0%	0	0	
TEZ.Migrations.OrgPassword	0	168	168	217	0%	0	0	
TEZ.Migrations.PaymentTicket	0	399	399	456	0%	0	0	
TEZ.Migrations.Payments	0	379	379	437	0%	0	0	
TEZ.Migrations.Seats	0	234	234	286	0%	0	0	
TEZ.Migrations.TestCreate	0	155	155	206	0%	0	0	
TEZ.Migrations.TezDbContextModelSnapshot	386	0	386	424	100%	0	0	
TEZ.Migrations.Ticket	0	278	278	331	0%	0	0	
TEZ.Migrations.routes	0	353	353	408	0%	0	0	

During white-box testing, we encountered issues with code coverage primarily due to test configuration errors, as shown in the attached screenshot. While we initially achieved decent manual coverage, automating the test cases led to a significant drop in coverage. Many test cases began throwing exceptions tied to configuration issues.

Despite repeated attempts over two days to resolve the problem, the underlying cause remained unclear, and the errors persisted. We believe this issue was beyond our control, likely rooted in deeper system or environment-level conflicts.

13. Work Division between group members

Roles:

Product Owner (PO): Ibtehaj Kazmi
Scrum Master (SM): Muaz Ahmed
Scrum Team: Usman Haroon, Muaz Ahmed, Ibtehaj Kazmi

Division of Work:

Ibtehaj Kazmi: Backend Development / Tester
Muaz Ahmed: Frontend Development
Usman Haroon: Database Management

14. Lesson learnt by group

Through this project, our group gained valuable hands-on experience with the Scrum model, learning how to effectively plan sprints, conduct daily stand-ups, and adapt to changing requirements using an agile mindset. We learned the importance of clear task distribution, time management, and continuous communication within the team. One major takeaway was the critical role of testing in software development. While we implemented white-box testing, configuration issues affected our code coverage, highlighting how even small misconfigurations can impact overall reliability. This emphasized the need for early testing integration, environment setup, and thorough validation processes. Overall, the project gave us a practical understanding of end-to-end software development—from ideation to deployment—while reinforcing the importance of adaptability, documentation, and collaboration in real-world projects.