TEZ 

ONE STOP SOLUTION FOR STUDENT TRANSPORT NEEDS

Members:

Ibtehaj Haider (22i-0767) - *PRODUCT OWNER*

Muaz Ahmed (22i-1125) - *SCRUM MASTER*

Usman Haroon (22i-1177) - *SCRUM MEMBER*

INTRODUCTION OF SYSTEM

Purpose:

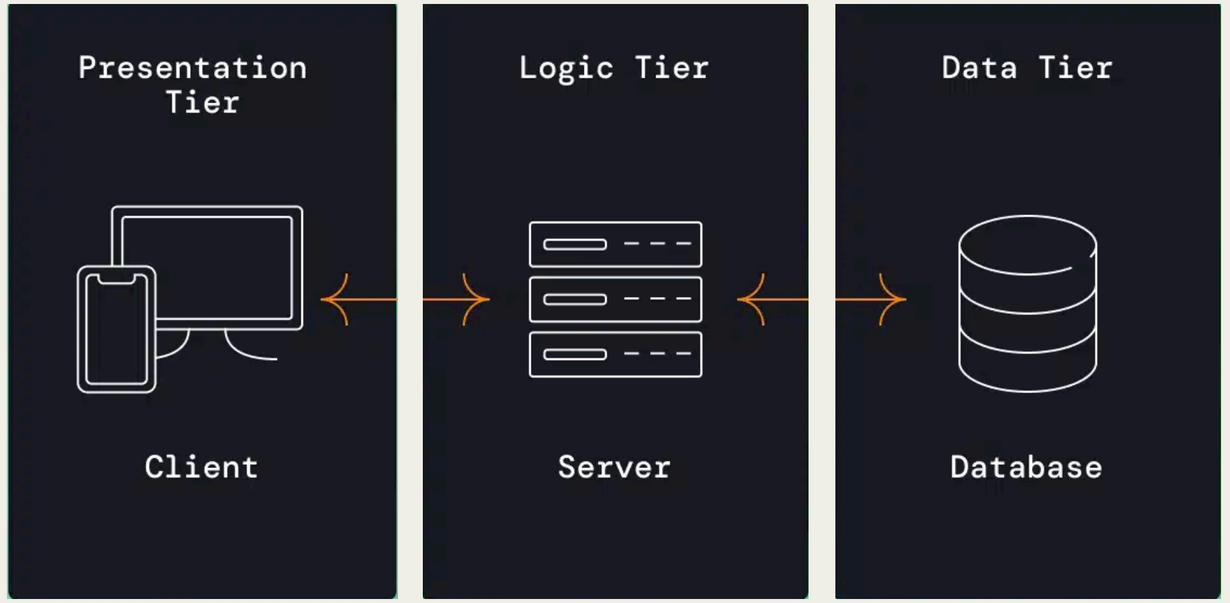
- Streamline and digitize the public transportation experience for educational institutions

Goals:

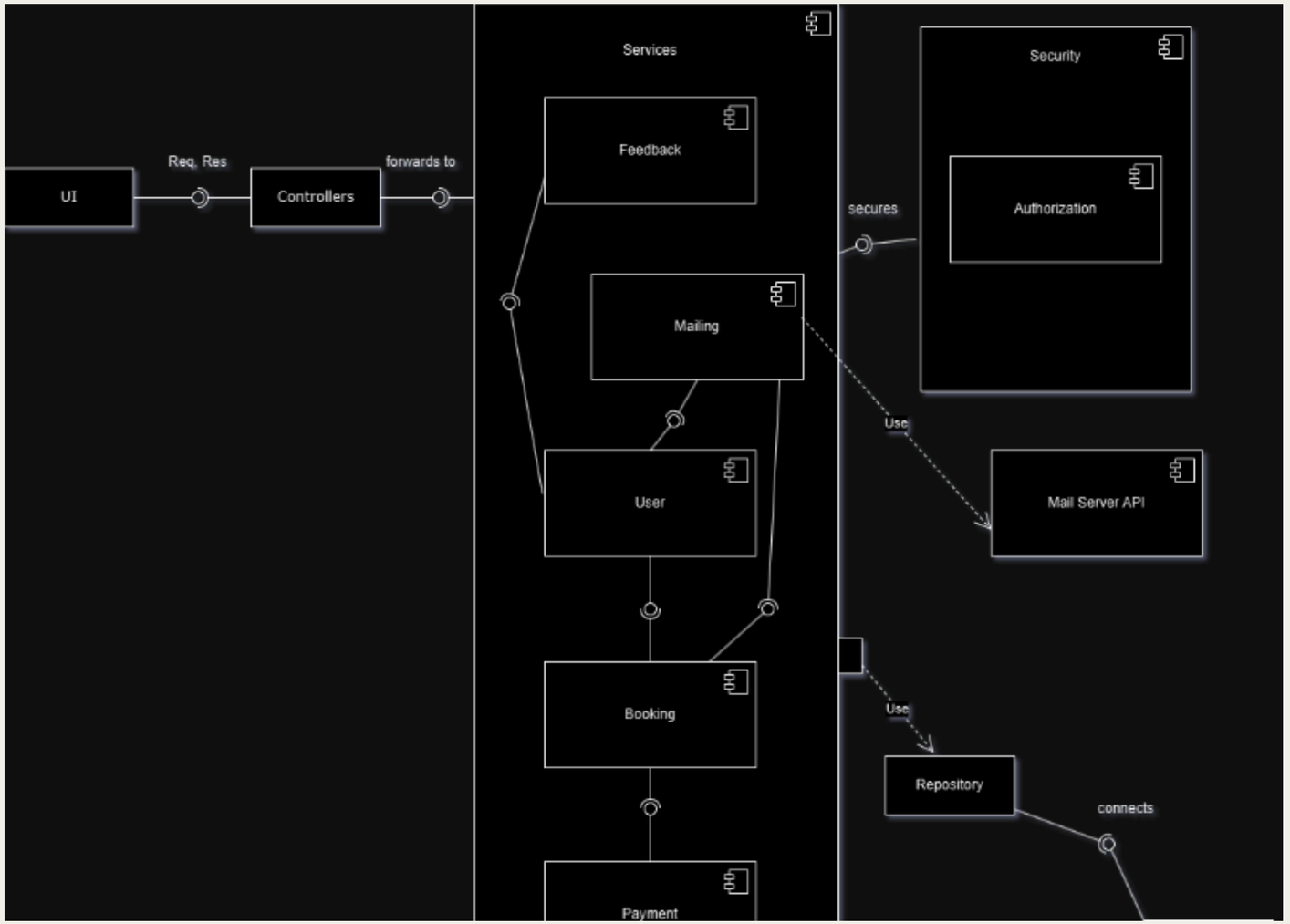
- Real-time bus tracking
- Automated seat management
- Secure authentication
- Improved communication between students, drivers, and administrators.

Architecture:

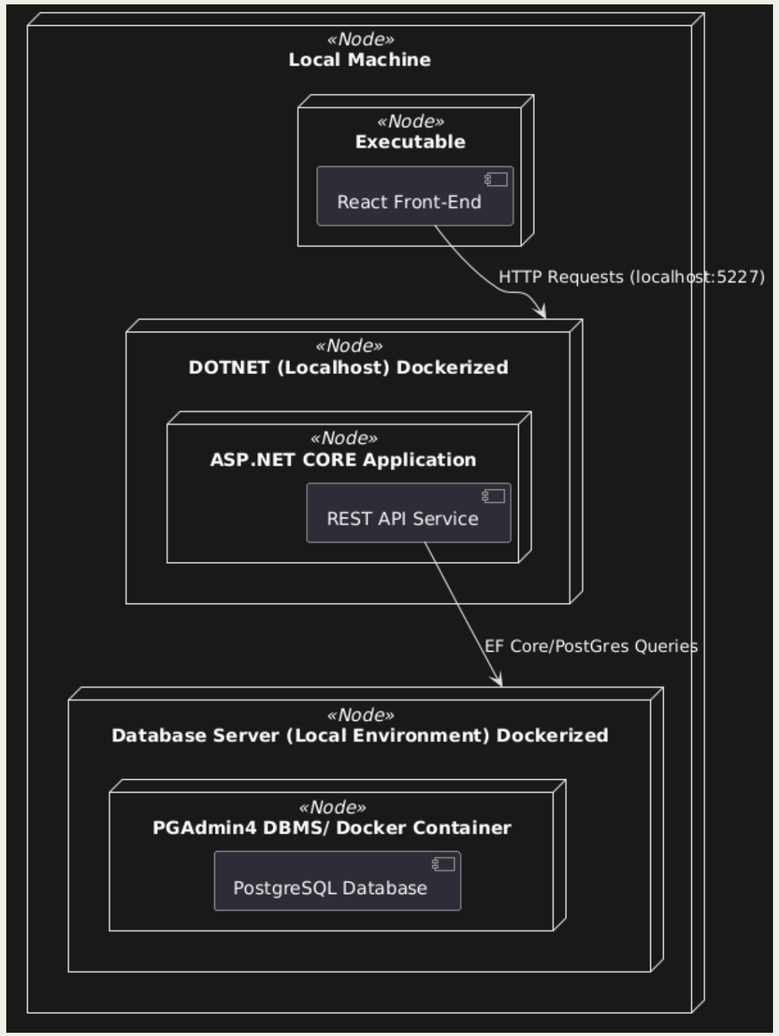
3- Tier Architecture



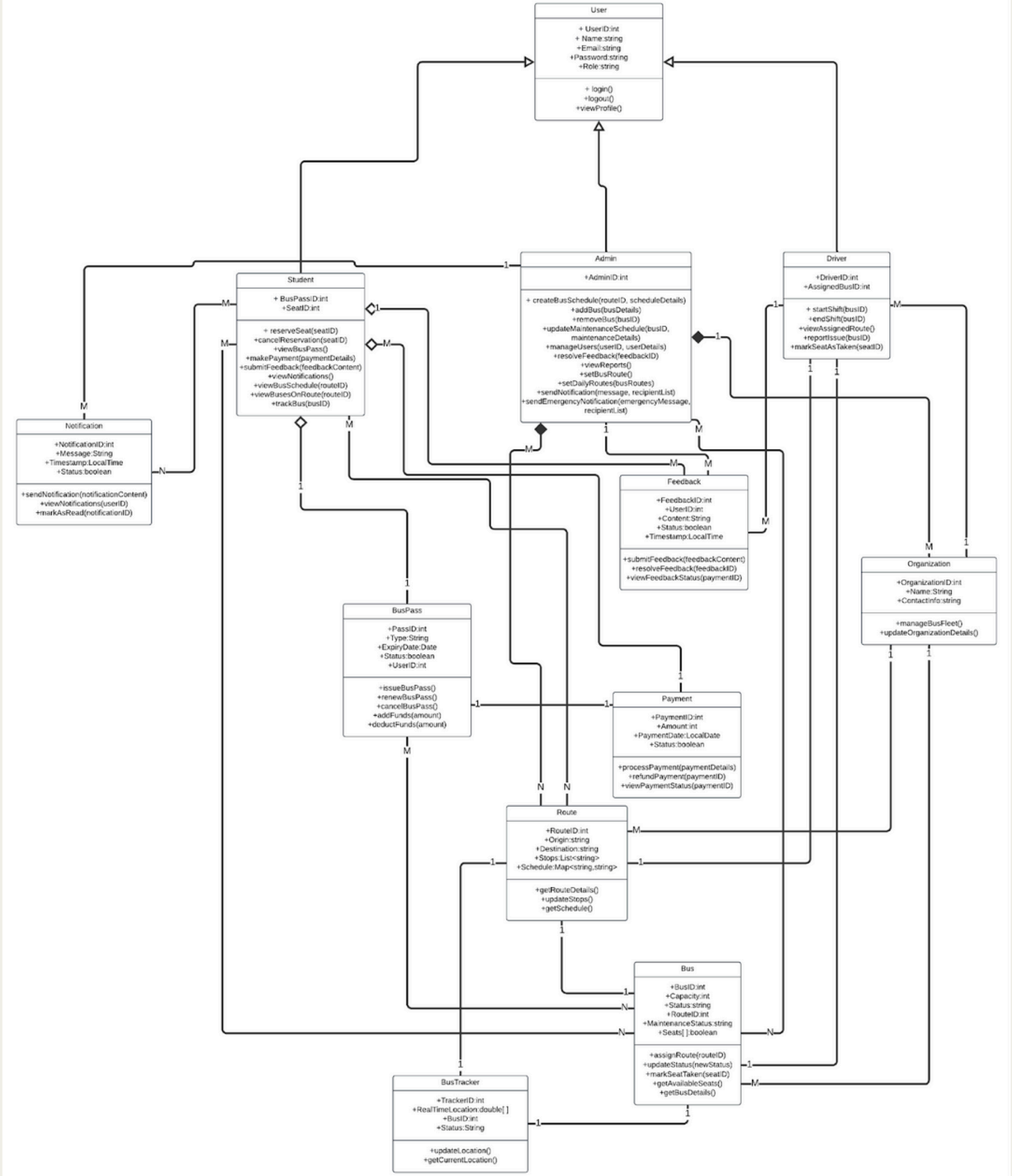
Component Diagram



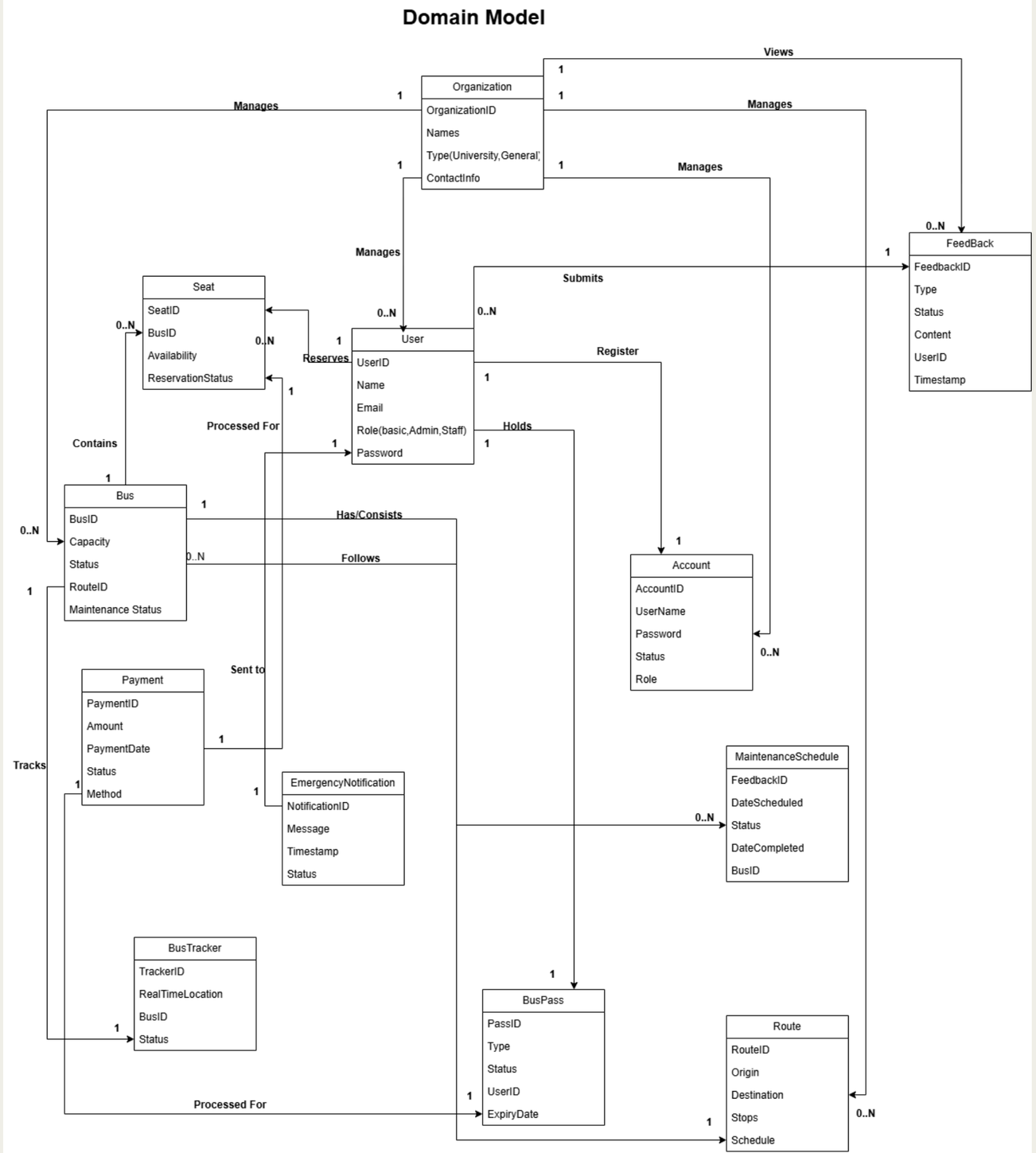
Deployment Diagram



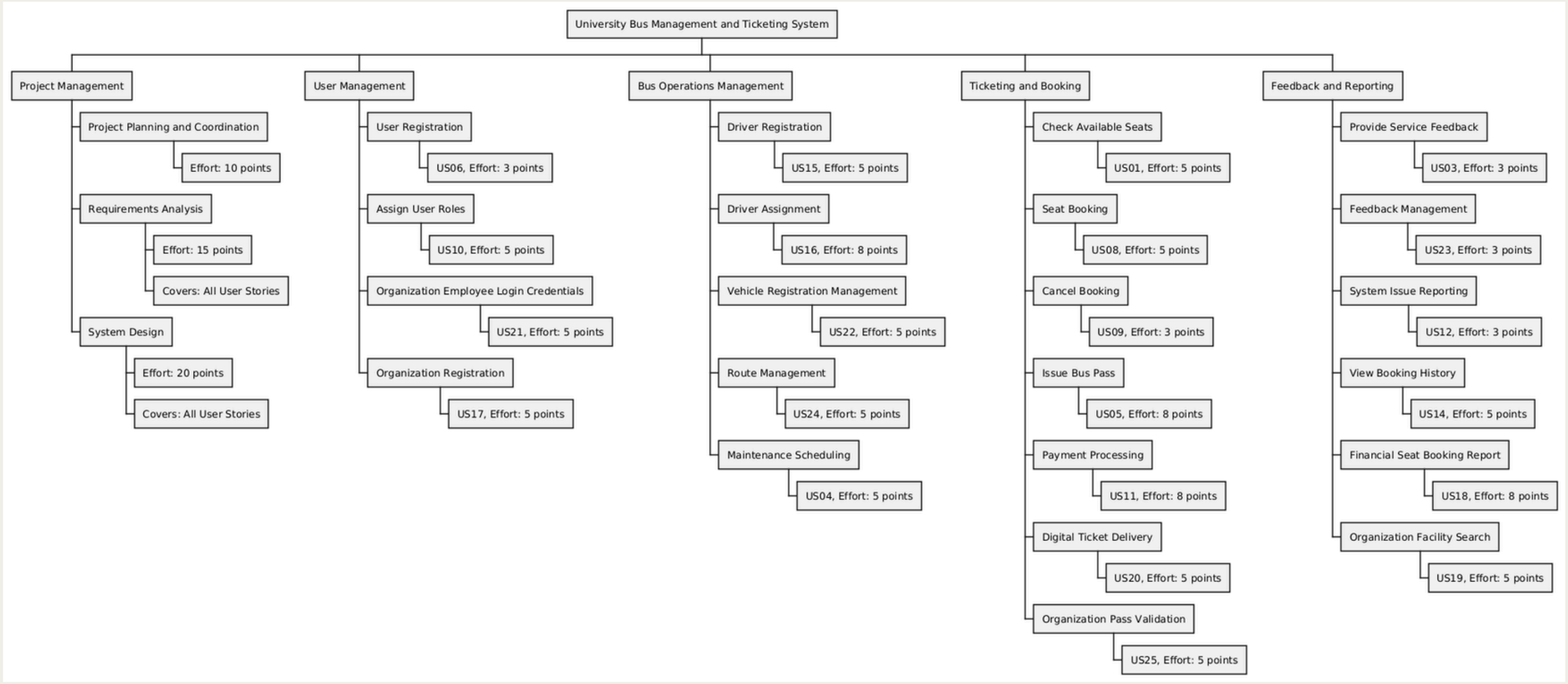
Class Diagram



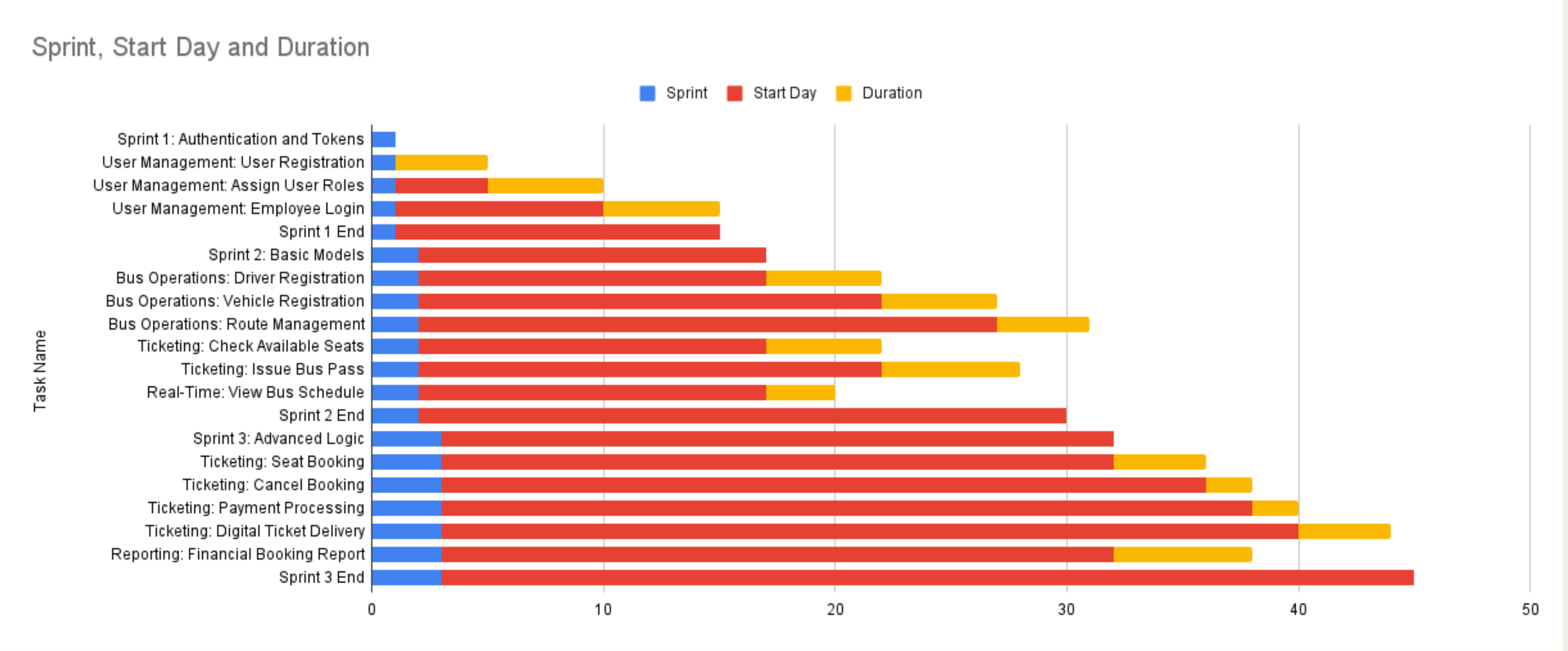
Domain Model



Project Planner - Work Breakdown Structure



Project Planner - Gantt Chart



User Stories - 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.

Importance: High
Estimate: 5 hours
Type: Search, Report/View

Acceptance Criteria:

- Given that I am a logged-in student, when I view the bus seat availability page, then I should see the current seat availability for all buses.
- Given that another user has just booked a seat, when I am viewing the seat availability page, then the seat status should automatically update to reflect the new booking.

User Stories - US02: View Bus Schedule

User Story: As a student, I want to view the bus schedule, so that I can plan my journey efficiently.

Importance: High
Estimate: 3 hours
Type: Report/View

Acceptance Criteria:

- Given that I am a logged-in student, when I navigate to the bus schedule page, then I should see an updated list of all bus routes with their departure and arrival times.
- Given that I am viewing the bus schedule, when I select a specific route, then I should see detailed expected arrival and departure times for all stops on that route.

User Stories - US03: Provide Service Feedback

User Story: As a student, I want to submit feedback on my bus experience, so that the administration can improve the service.

Importance: Medium
Estimate: 3 hours
Type: Manage Data

Acceptance Criteria:

- Given that I am a logged-in student, when I navigate to the feedback section, then I should see a form to submit my feedback about the bus service.
- Given that I have filled out the feedback form, when I submit the form, then I should receive a confirmation message that my feedback has been stored in the system.

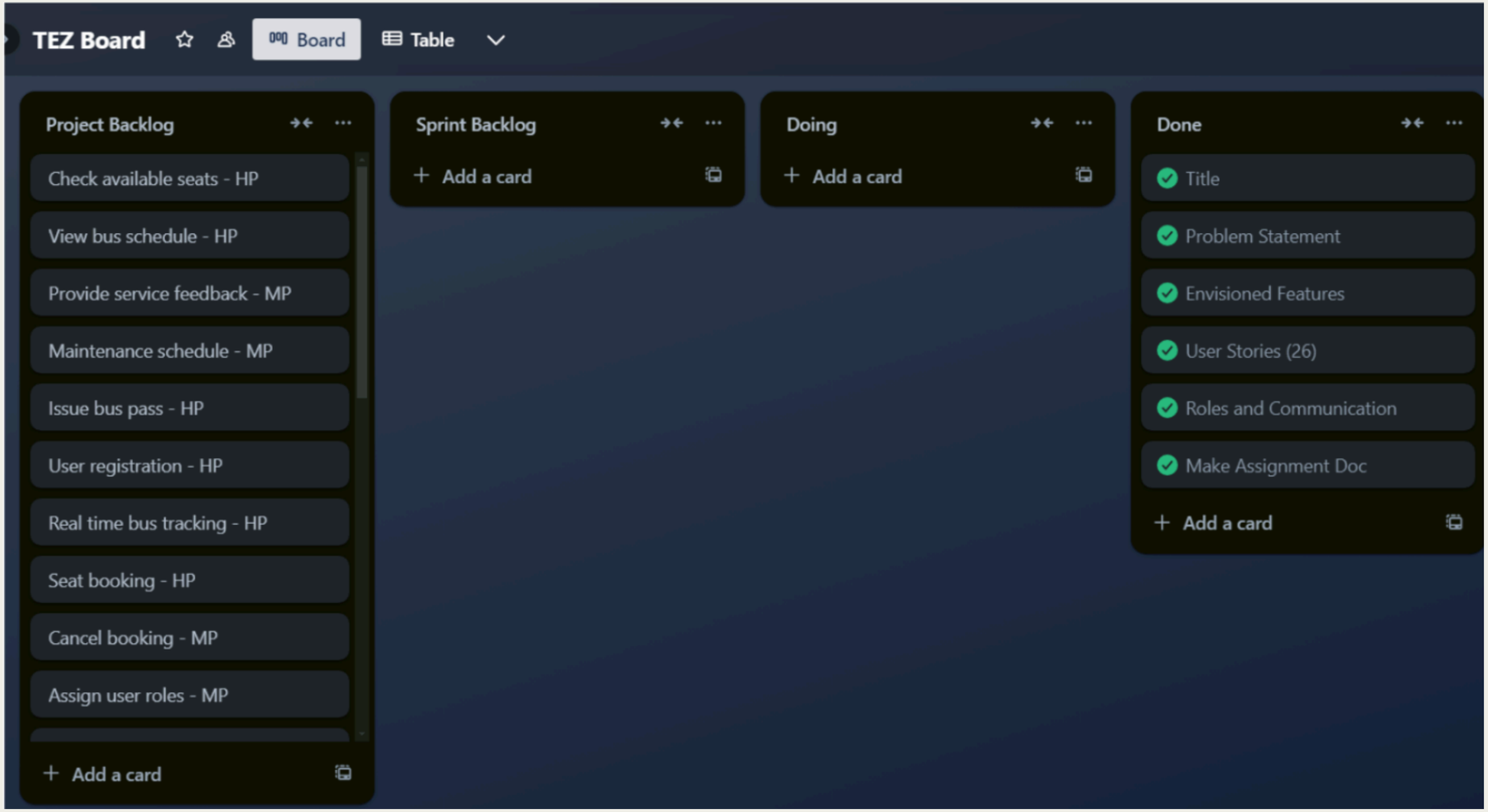
User Stories - US04: Seat Booking

User Story: As a student, I want to reserve a seat in advance, so that I have a confirmed spot on the bus.

Importance: High
Estimate: 5 hours
Type: Workflow, Manage Data

Acceptance Criteria:

- Given that I am a logged-in student, when I navigate to the seat booking page, then I should see all available seats that can be reserved.
- Given that I select an available seat, when I confirm my booking, then I should receive a confirmation of my reservation.
- Given that a seat has been reserved, when another student views the seat availability, then the reserved seat should be shown as unavailable.

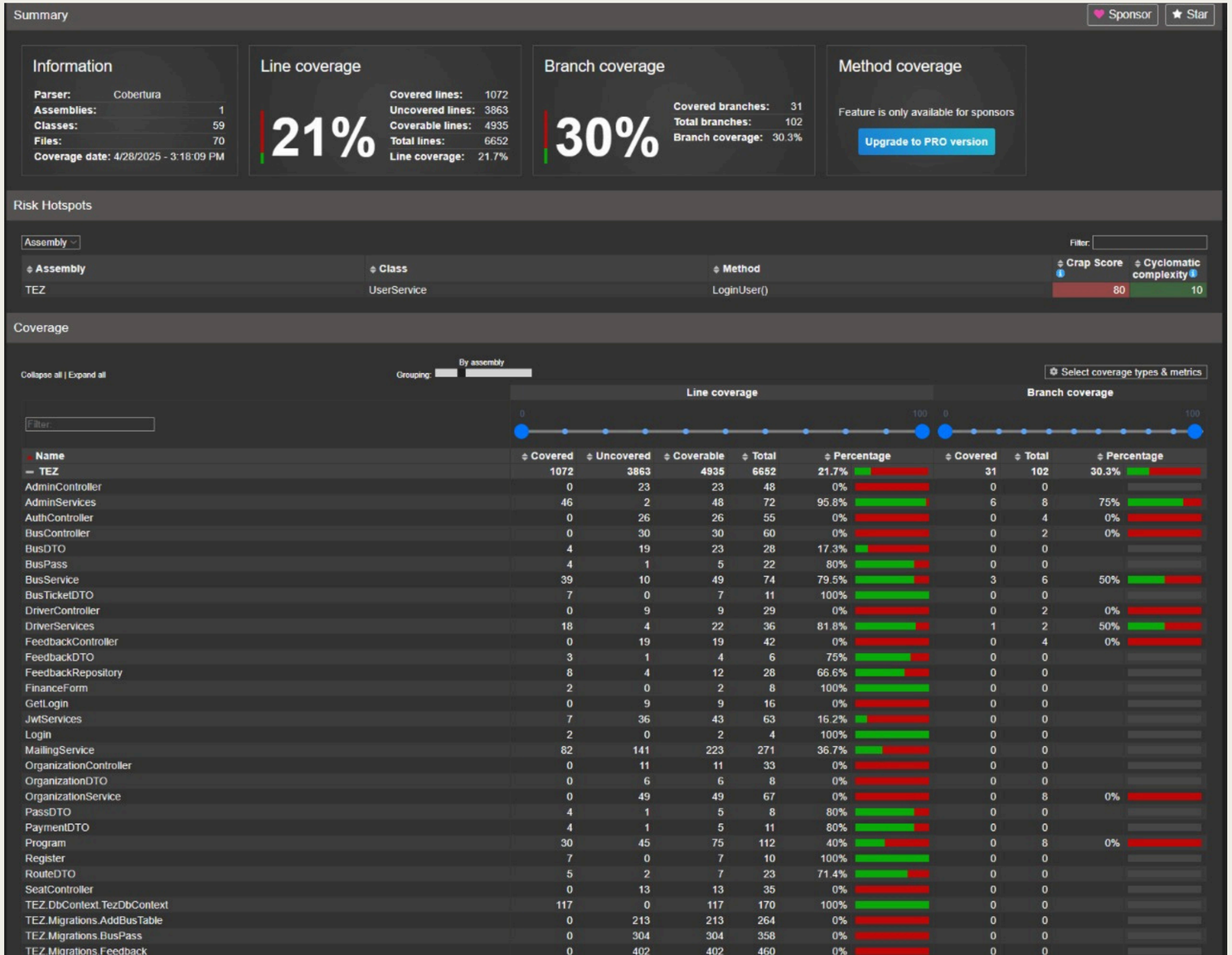

TEZ Board
Board
Table
Project Backlog
Check available seats - HP
View bus schedule - HP
Provide service feedback - MP
Maintenance schedule - MP
Issue bus pass - HP
User registration - HP
Real time bus tracking - HP
Seat booking - HP
Cancel booking - MP
Assign user roles - MP
Add a card
Sprint Backlog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

White Box Testing.

```
[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);
}
```

```
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
}
```

White Box Testing - Coverage

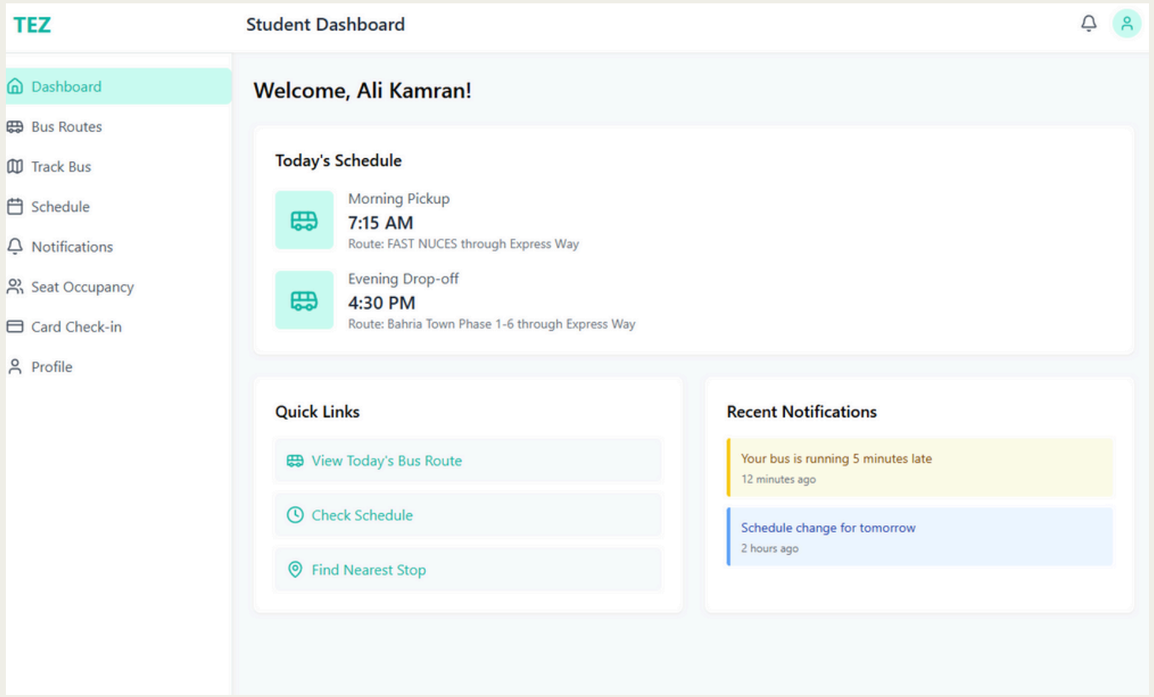


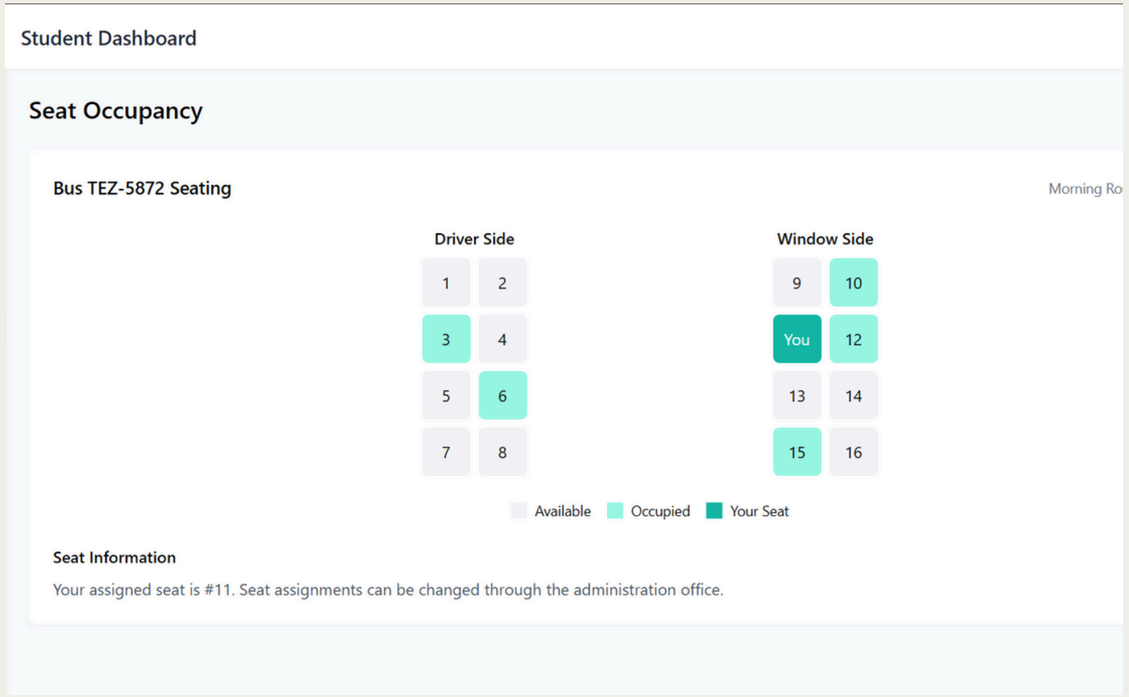
Black Box Testing

```
[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]
```

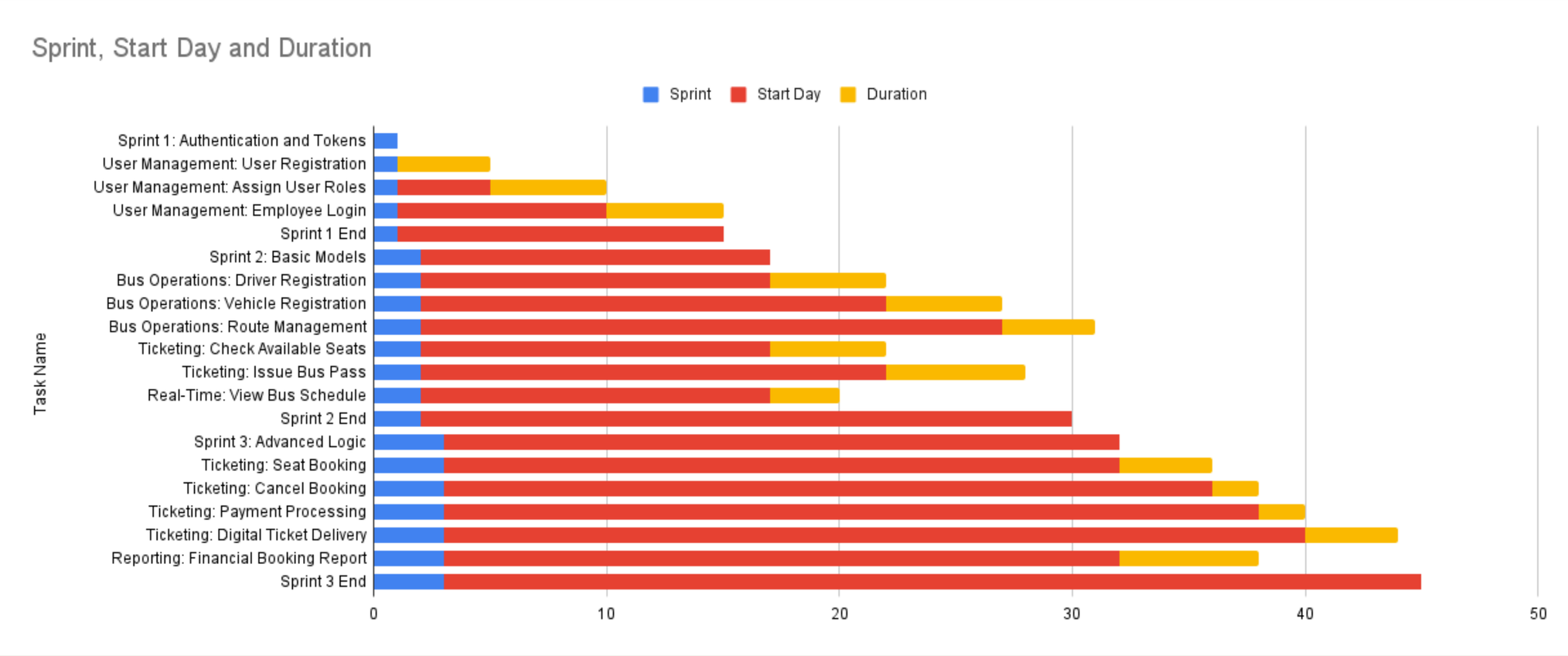
```
0 references
public async Task GetAllUsers_ReturnsListOfUsers()
{
    // Arrange
    await _fixture.SeedTestData();

    // 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();
}
```

Design and Implementation





Burndown Chart



Further Improvements

- AI Route Optimization
- Digital Bus Pass
- Driver Rating

Lessons Learnt

- **Scrum Model**
  - Sprints, Daily stand-up, Agile environment
- **Team Work**
  - Task Distribution, Time management, Communication
- **Importance of Testing**

Thank you!

QUESTIONS?

molarmuaz.github.io/tez