SOFTWARE ENGINEERING CONCEPTS LAB MANUAL

SOFTWARE CONCEPTS & ENGINEERING

LAB MANUAL

Overview of the project -

Business Architecture Diagram

Requirements as User Stories

Architecture Diagram Depicting The

Test Strategy

Deployment Architecture of the application

Overview of the Project: Timetable Management System for Teachers

Problem:

Inefficient scheduling processes: Traditional methods for creating and managing timetables can be time-consuming and prone to errors. This often results in conflicts, overlaps, and inefficiencies.

Manual updates and communication: Changes to schedules need to be communicated manually, leading to delays and potential miscommunications.

Limited accessibility: Teachers and administrators may find it difficult to access and update timetable information on-the-go, especially when relying on paper-based or isolated digital systems.

Data-driven Approach:

This project will leverage data to:

Automate scheduling: Use algorithms to automatically generate optimal timetables based on teachers' availability, course requirements, and classroom resources.

Conflict resolution: Identify and resolve scheduling conflicts proactively, ensuring a smooth and efficient timetable management process.

Dynamic updates: Allow real-time updates to timetables, instantly notifying all affected parties about any changes.

Resource allocation: Analyze data to optimize the use of classrooms, equipment, and other resources, minimizing downtime and maximizing utility.

Benefits for Users:

Time-saving: Automate the creation and management of timetables, freeing up time for teachers and administrators to focus on other important tasks.

Improved accuracy: Reduce scheduling errors and conflicts through intelligent automation and conflict resolution mechanisms.

Enhanced accessibility: Provide teachers and administrators with easy access to up-to-date timetables via mobile and web applications.

Increased flexibility: Facilitate quick and efficient updates to schedules, ensuring all stakeholders are informed in real-time.

Integration and Collaboration:

Interdepartmental coordination: Integrate with other school systems (e.g., student information systems, resource management platforms) to ensure seamless data flow and coordination.

User-friendly interface: Develop an intuitive and easy-to-use interface that allows teachers and administrators to interact with the system effortlessly.

Feedback and improvement: Implement features to collect user feedback and continuously improve the system based on user needs and preferences.

Secure data management: Ensure robust data security measures to protect sensitive timetable and personal information.

Conclusion:

The Timetable Management System for Teachers aims to modernize and streamline the scheduling process in educational institutions. By leveraging data and automation, the system will provide significant benefits in terms of time savings, accuracy, and accessibility, while fostering a more collaborative and flexible environment for both teachers and administrators.

Business Architecture Diagram

Business Need:

TeachTime aims to address the inefficiencies and complexities in managing teachers' timetables in schools and colleges. These issues include:

- Manual Scheduling: Creating and adjusting timetables manually is time-consuming and prone to errors.
- Conflict Management: Avoiding schedule conflicts without a centralized system is challenging.
- Limited Accessibility: Teachers may not have easy access to the most up-to-date timetable information.

Current Process (as-is):

- Administrators: Manually create and adjust timetables using spreadsheets or paper-based systems.
- Distribution: Share the timetables through email or physical copies posted on notice boards.
- Teachers: Check their email or notice boards regularly to stay updated on their schedules.
- Adjustments: Communicate changes through additional emails or verbal announcements, leading to confusion and missed updates.

Business Problems:

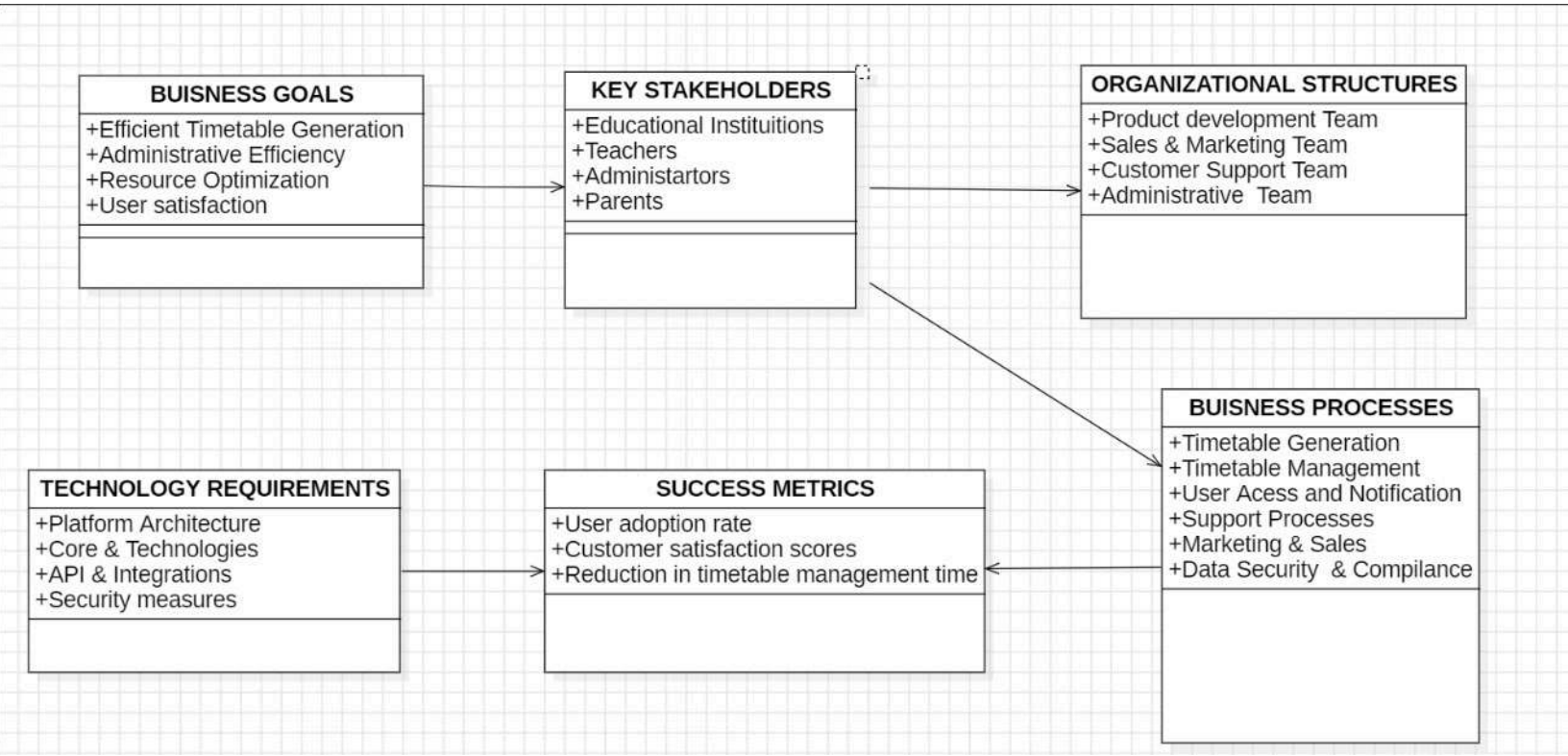
- Inefficiencies: Manual timetable creation and adjustments are time-consuming and error-prone.
- Schedule Conflicts: Without a centralized system, avoiding and managing schedule conflicts is difficult.
- Accessibility Issues: Teachers may not have timely access to the most current timetable information, leading to confusion and missed classes.

Solution:

TeachTime provides a centralized platform for efficient and error-free timetable management, ensuring that teachers and administrators can easily access and update scheduling information. This translates to:

- Increased Efficiency: Automated scheduling and adjustments save time and reduce errors.
- Conflict Resolution: A centralized system helps prevent and quickly resolve scheduling conflicts.
- Improved Accessibility: Real-time access to updated timetables ensures that teachers are always informed about their schedules.
- Enhanced Communication: Notifications and alerts keep teachers updated on any changes to their timetable.

Business Architecture Diagram



Software Architecture Diagram

Architecture Diagram

Model-View-Controller (MVC) Pattern

Model

The Model represents the data and the business logic of the timetable management system. It handles the retrieval, storage, and manipulation of timetable data.

Key Components:

1. **Teacher Model: Handles teacher information including personal details and availability.**
2. **Timetable Model: Manages the timetable entries including class schedules, subjects, and assigned teachers.**
3. **Classroom Model: Stores data about classrooms and their availability.**
4. **Subject Model: Contains information about the subjects being taught.**
5. **Conflict Detection Logic: Ensures no scheduling conflicts exist (e.g., double-booking a teacher or a classroom).**

View

The View is responsible for presenting data to the user. It displays the timetable and provides interfaces for viewing and interacting with the timetable data.

Key Components:

1. **Timetable View: Displays the timetable to teachers and administrators.**
2. **Teacher View: Shows individual teacher schedules.**
3. **Classroom View: Displays availability and schedules for classrooms.**
4. **lnput Forms: Allows users to input data for new timetables, update availability, etc.**

Controller

The Controller handles user input and updates the Model and View accordingly. It processes requests, manipulates data, and determines which view to display.

Key Components:

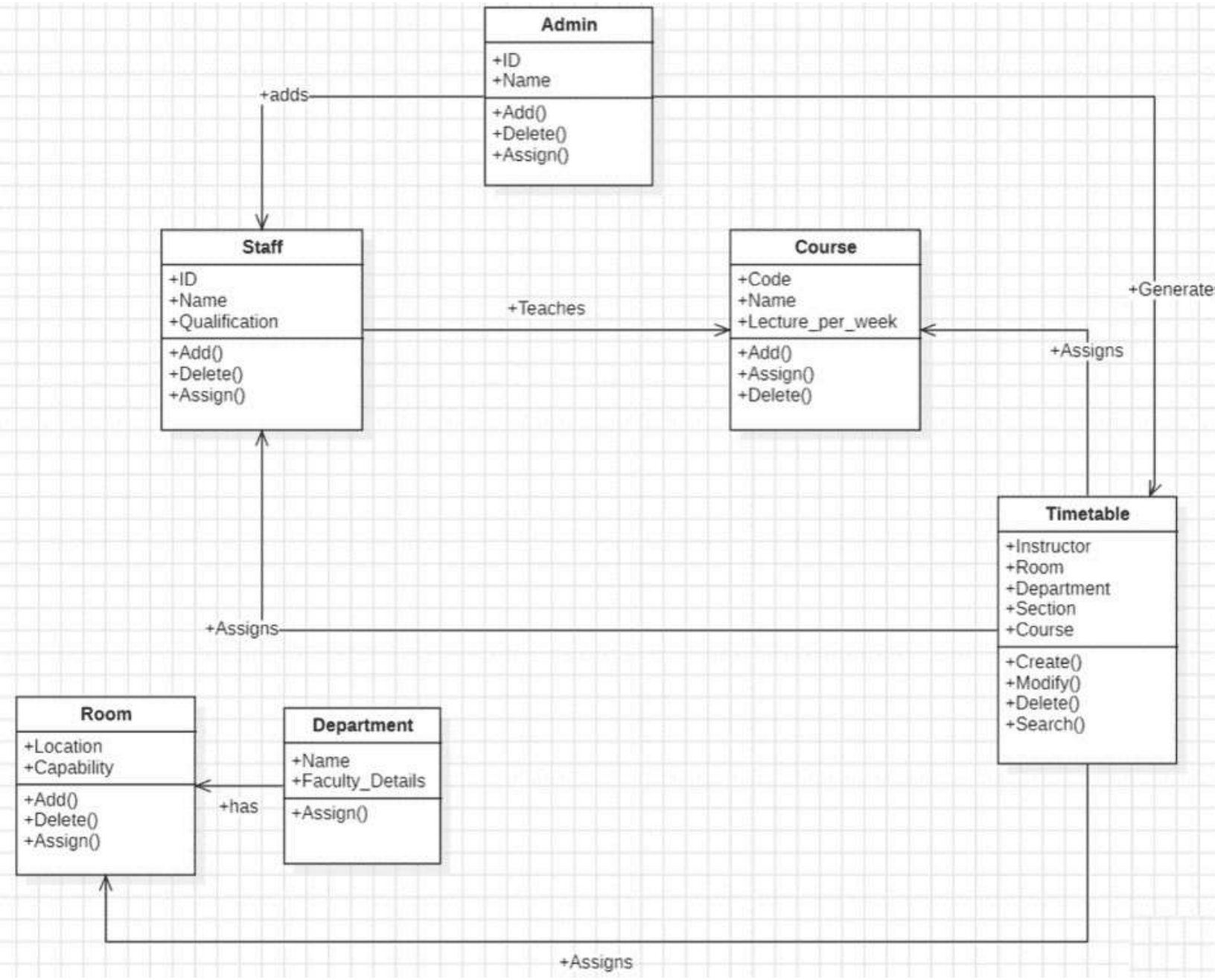
1. **Timetable Controller: Manages requests related to timetable creation, updates, and deletions.**
2. **Teacher Controller: Handles requests related to teacher information and availability.**
3. **Classroom Controller: Manages requests related to classroom data and availability.**
4. **Subject Controller: Handles requests related to subjects.**

Summary

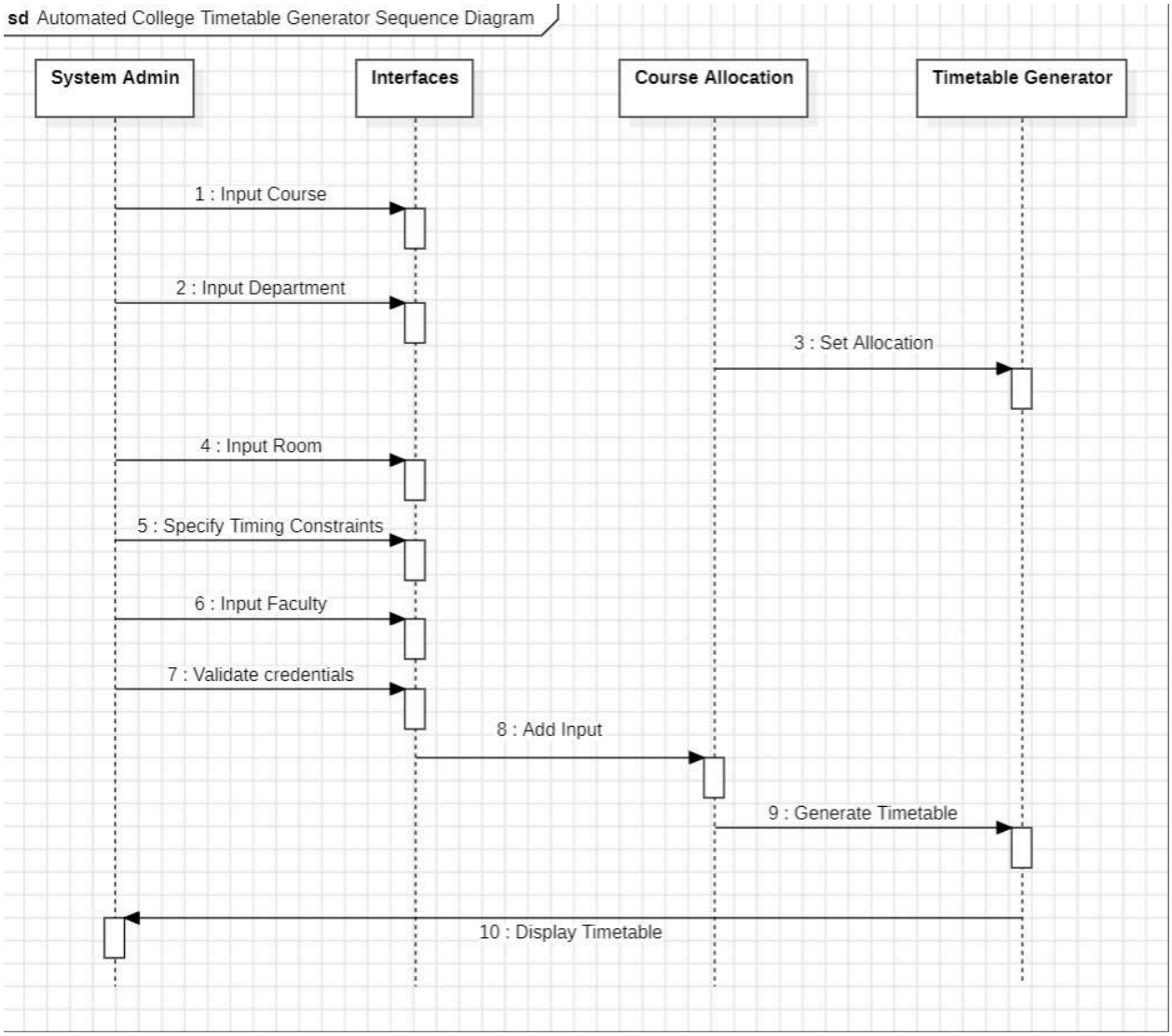
In the MVC pattern for the timetable management system:

- **Model: Manages the core data and business logic.**
- **View: Renders the timetable and related information to the users.**
- **Controller: Acts as an intermediary, handling user inputs and updating the Model and View.**

- **Class Diagram**

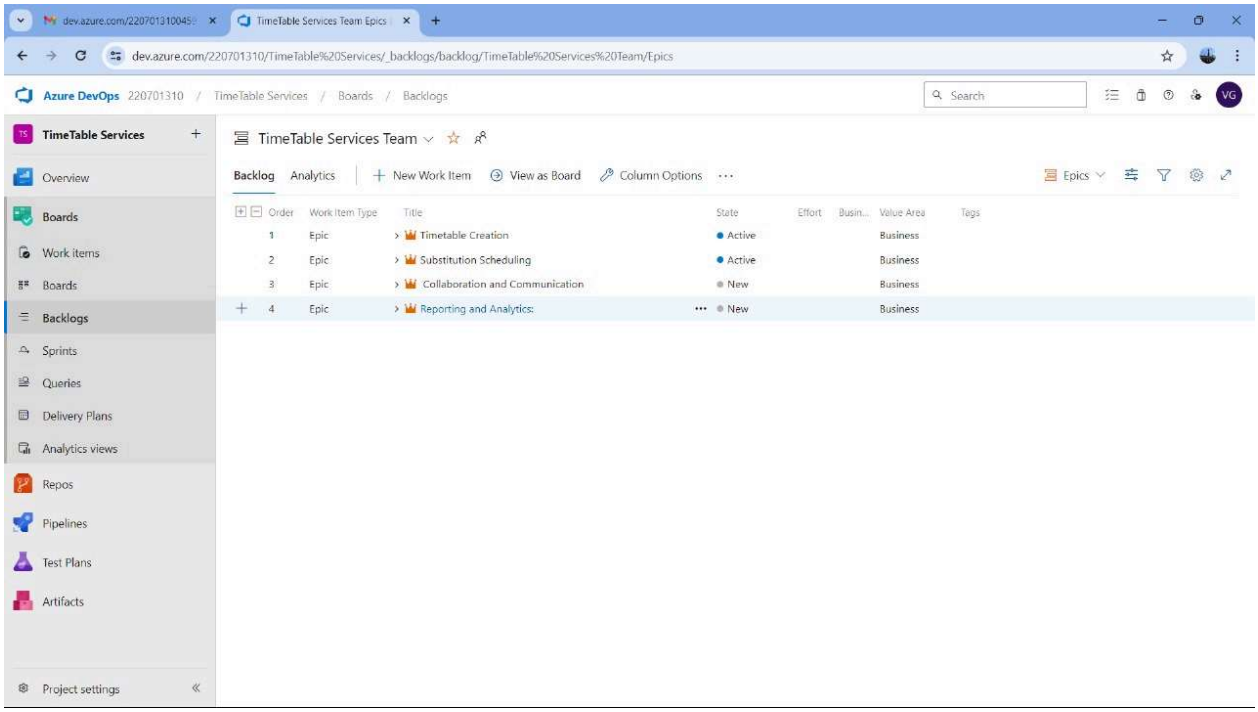


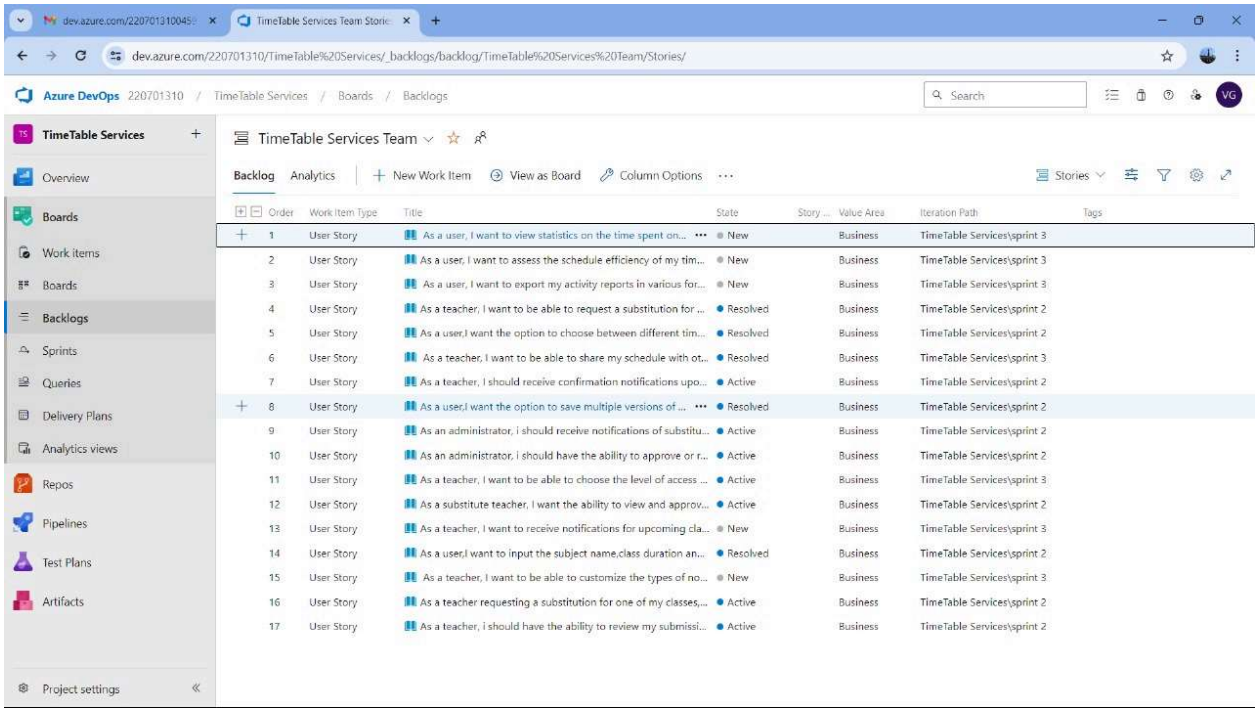
- **Sequence Diagram**

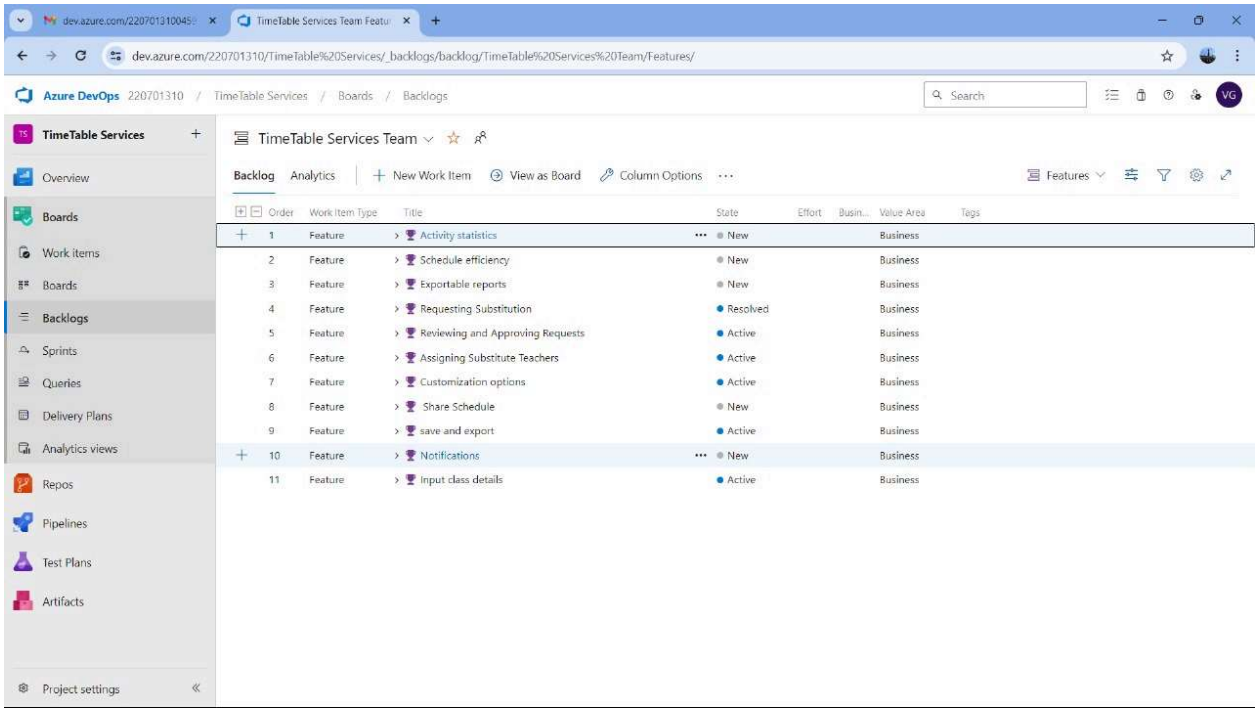


Requirements as User Stories

- **User Story 1 :** As a user , I want the option to choose between different timetable layouts so that I can find layout that best suits my preferences and needs. (This user story primarily focuses on choosing preferred timetable layout)
- **User Story 2:** As a user , I want the option to save multiple versions of timetable with different adjustments so,that I can compare and choose the most suitable one for my schedule. (This user story primarily focuses on saving multiple versions of my timetable)
- **User Story 3 :**As a user,I want to input the subject name,class duration and preffered time slots for each of my classes ,so that timetable generator can schedule them efficiently .(This primarily focuses on to generate time table with available inputs)
- **User Story 4:** As a teacher, I want to be able to request a substitution for one of my classes if I am unable to teach.(This primarily focuses on substitution period)
- **User Story 5:**As a teacher requesting a substitution for one of my classes, I want the application to provide priority recommendations for substitute teachers based on their availability and relevance to the class or subject. (This primarily focuses on application to recommend substitute teachers based on their availability)
- **User Story 6**:As an administrator, I should have the ability to approve or reject substitution requests based on various factors, such as validity and urgency.(This primarily focuses on ability to approve or reject substitution requests)
- **User Story 7**: As a teacher, I want to be able to share my schedule with other teachers or staff members.(This primarily focuses on sharing my schedule)
- **User Story 8:**As a teacher, I want to receive notifications for upcoming classes or schedule changes.(This primarily focuses on reminder system)
- **User Story 9:** As a user, I want to view statistics on the time spent on different activities within my timetable, so that I can analyze my productivity and identify areas for improvement.(This primarily focuses on self improvement)
- **User story 10:** As a user, I want to export my activity reports in various formats (e.g., PDF, CSV), so that I can share them with others or archive them for future reference.(This primarily focuses on sharing reports)


dev.azure.com/220701310045
TimeTable Services Team Epics
dev.azure.com/220701310/TimeTable%20Services/_backlogs/backlog/TimeTable%20Services%20Team/Epics
Azure DevOps 220701310 / TimeTable Services / Boards / Backlogs
Search
VG
TimeTable Services
Overview
Boards
Work items
Boards
Backlogs
Sprints
Queries
Delivery Plans
Analytics views
Repos
Pipelines
Test Plans
Artifacts
Project settings
TimeTable Services Team
Backlog
Analytics
New Work Item
View as Board
Column Options
Epics
Order
Work Item Type
Title
State
Effort
Busin...
Value Area
Tags
1 Epic Timetable Creation Active Business
2 Epic Substitution Scheduling Active Business
3 Epic Collaboration and Communication New Business
4 Epic Reporting and Analytics: New Business


dev.azure.com/220701310/TimeTable%20Services/_backlogs/backlog/TimeTable%20Services%20Team/Stories/
Azure DevOps 220701310 / TimeTable Services / Boards / Backlogs
Search
VG
TimeTable Services
Overview
Boards
Work items
Boards
Backlogs
Sprints
Queries
Delivery Plans
Analytics views
Repos
Pipelines
Test Plans
Artifacts
Project settings
TimeTable Services Team
Backlog
Analytics
New Work Item
View as Board
Column Options
Stories
Order
Work Item Type
Title
State
Story ...
Value Area
Iteration Path
Tags
1 User Story As a user, I want to view statistics on the time spent on... New Business TimeTable Services\sprint 3
2 User Story As a user, I want to assess the schedule efficiency of my tim... New Business TimeTable Services\sprint 3
3 User Story As a user, I want to export my activity reports in various for... New Business TimeTable Services\sprint 3
4 User Story As a teacher, I want to be able to request a substitution for ... Resolved Business TimeTable Services\sprint 2
5 User Story As a user,I want the option to choose between different tim... Resolved Business TimeTable Services\sprint 2
6 User Story As a teacher, I want to be able to share my schedule with ot... Resolved Business TimeTable Services\sprint 3
7 User Story As a teacher, I should receive confirmation notifications upo... Active Business TimeTable Services\sprint 2
8 User Story As a user,I want the option to save multiple versions of ... Resolved Business TimeTable Services\sprint 2
9 User Story As an administrator, i should receive notifications of substitu... Active Business TimeTable Services\sprint 2
10 User Story As an administrator, i should have the ability to approve or r... Active Business TimeTable Services\sprint 2
11 User Story As a teacher, I want to be able to choose the level of access ... Active Business TimeTable Services\sprint 3
12 User Story As a substitute teacher, I want the ability to view and approv... Active Business TimeTable Services\sprint 2
13 User Story As a teacher, I want to receive notifications for upcoming cla... New Business TimeTable Services\sprint 3
14 User Story As a user,I want to input the subject name,class duration an... Resolved Business TimeTable Services\sprint 2
15 User Story As a teacher, I want to be able to customize the types of no... New Business TimeTable Services\sprint 3
16 User Story As a teacher requesting a substitution for one of my classes,... Active Business TimeTable Services\sprint 2
17 User Story As a teacher, i should have the ability to review my submissi... Active Business TimeTable Services\sprint 2



Non Functional Requirements

1. Usability:
    - Friendly User-Interface
    - Accessibility
2. Maintainability:
    - Modular Design
    - Documentation
    - Robustness
3. Performance:
    - Scalability
    - Response Time
    - Availability
4. Security:
    - Authentication and Authorization
    - Data Security

Poker Planning Methodology

Poker Planning is a collaborative estimation technique used in Agile projects to estimate the effort or complexity of user stories.

User Stories

1. User Story 1: Choosing Timetable Layouts
    - Poker Planning Points: 3
2. User Story 2: Saving Multiple Timetable Versions
    - Poker Planning Points: 5
3. User Story 3: Inputting Subject Details for Timetable Generation
    - Poker Planning Points: 8
4. User Story 4: Requesting Substitution
    - Poker Planning Points: 5
5. User Story 5: Recommending Substitute Teachers
    - Poker Planning Points: 8
6. User Story 6: Approving or Rejecting Substitution Requests
    - Poker Planning Points: 5
7. User Story 7: Sharing Schedule with Others
    - Poker Planning Points: 3
8. User Story 8: Receiving Schedule Notifications
    - Poker Planning Points: 3
9. User Story 9: Viewing Time Spent on Activities
    - Poker Planning Points: 8
10. User Story 10: Exporting Activity Reports
    - Poker Planning Points: 5

Non Functional Requirements

1. Usability:
    - Friendly User-Interface: 3
    - Accessibility: 5
2. Maintainability:
    - Modular Design: 3

    - Documentation: 2
    - Robustness: 4
3. Performance:
    - Scalability: 6
    - Response Time: 5
    - Availability: 7
4. Security:
    - Authentication and Authorization: 6
    - Data Security: 7

Test plans

User story 1: User Login

Description: as a user, I want to log in so that I can access the timetable management system.

Happy path test cases:

Test case 1:

- log in with valid credentials.
- Navigate to the login page.

- Enter a valid username and password.
- Click the "login" button. Verify that the user is redirected to the dashboard.

Expected result: user is successfully logged in and redirected to the dashboard.

Error scenarios:

Test case 2:

- log in with invalid credentials.
- Navigate to the login page.
- Enter an invalid username and/or password.
- Click the "login" button. Verify that an appropriate error message is displayed.

Expected result: error message indicates invalid login credentials.

Test case 3:

- attempt to log in with empty fields.
- Navigate to the login page.
- Leave the username and password fields empty.
- Click the "login" button. Verify that an appropriate error message is displayed.

Expected result: error message indicates that fields cannot be empty.

User story 2: Alert Notification

Description: as a user, I want to receive alert notifications for timetable changes so that I am aware of updates.

Happy path test cases:

Test case 1:

- receive an alert notification for a timetable update.
- Admin updates a timetable.
- User receives a notification about the update.
- Verify that the notification contains the correct update details.

Expected result: user receives a timely and accurate alert notification.

Error scenarios:

Test case 2:

- notification delivery failure.
- Admin updates a timetable.
- Notification system fails to send an alert.
- Verify that an error message is logged.

Expected result: error message indicates notification delivery failure.

User story 3: Timetable Creation

Description: as an admin, I want to create a new timetable so that i can manage the schedule for classes.

Happy path test cases:

Test case 1:

- create a new timetable with valid details.
- Navigate to the "create timetable" page.
- Enter valid timetable details (e.g., name, dates, times).
- Click the "save" button. Verify the timetable is created successfully and displayed in the timetable list.

  Expected result: timetable is created and displayed correctly.

Error scenarios:

Test case 2:

- attempt to create a timetable with missing required fields.
- Navigate to the "create timetable" page.
- Leave required fields (e.g., name) empty. Click the "save" button. Verify that the appropriate error message is displayed.

  Expected result: error message indicates that required fields are missing.

User story 4: Timetable Modification

Description: as an admin, I want to edit an existing timetable so that i can update schedules.

Happy path test cases:

Test case 1:

- edit a timetable with valid details.

- Navigate to the "timetable list" page.
- Select a timetable to edit. Update the timetable details.
- Click the "save" button. Verify the timetable is updated successfully.

  Expected result: timetable is updated and displayed correctly.

Error scenarios:

Test case 2:

- attempt to edit a timetable and remove required fields.
- Navigate to the "timetable list" page.
- Select a timetable to edit. Remove required fields.
- Click the "save" button. Verify that the appropriate error message is displayed.

  Expected result: error message indicates that required fields cannot be empty.

User story 5: Teacher Substitution

Description: as an admin, I want to manage teacher substitutions so that classes can continue without interruption.

Happy path test cases:

Test case 1:

- substitute a teacher for a scheduled class.
- Navigate to the "substitute teacher" page.
- Select the class and the current teacher.

- Choose a substitute teacher from the list.
- Click the "confirm" button. Verify that the substitution is reflected in the timetable.

  Expected result: the timetable is updated to show the substitute teacher for the class.

Error scenarios:

Test case 2:

- attempt to substitute a teacher with invalid details.
- Navigate to the "substitute teacher" page.
- Select a class and current teacher.
- Choose an invalid substitute teacher (e.g., one not available).
- Click the "confirm" button. Verify that an appropriate error message is displayed.

  Expected result: error message indicates invalid substitution details.

Test case 3:

- attempt to substitute a teacher when no teachers are available.
- Navigate to the "substitute teacher" page.
- Select a class and current teacher.
- Verify that no substitute teachers are available in the list.
- Click the "confirm" button. Verify that an appropriate error message is displayed.

  Expected result: error message indicates no available substitute teachers.

GitHub Repository Structure

1. Project Structure Overview:

2. **Naming Conventions**:

- **Packages**: Follow a standard reverse domain naming convention (e.g., com.example.timetable).
- **Classes**: Use CamelCase for class names (e.g., TimetableApplication, TimetableController).
- **Methods**: Use camelCase for method names (e.g., createTimetable, deleteTimetable).
- **Files and Folders**: Use lowercase with hyphens for filenames and folders when applicable.

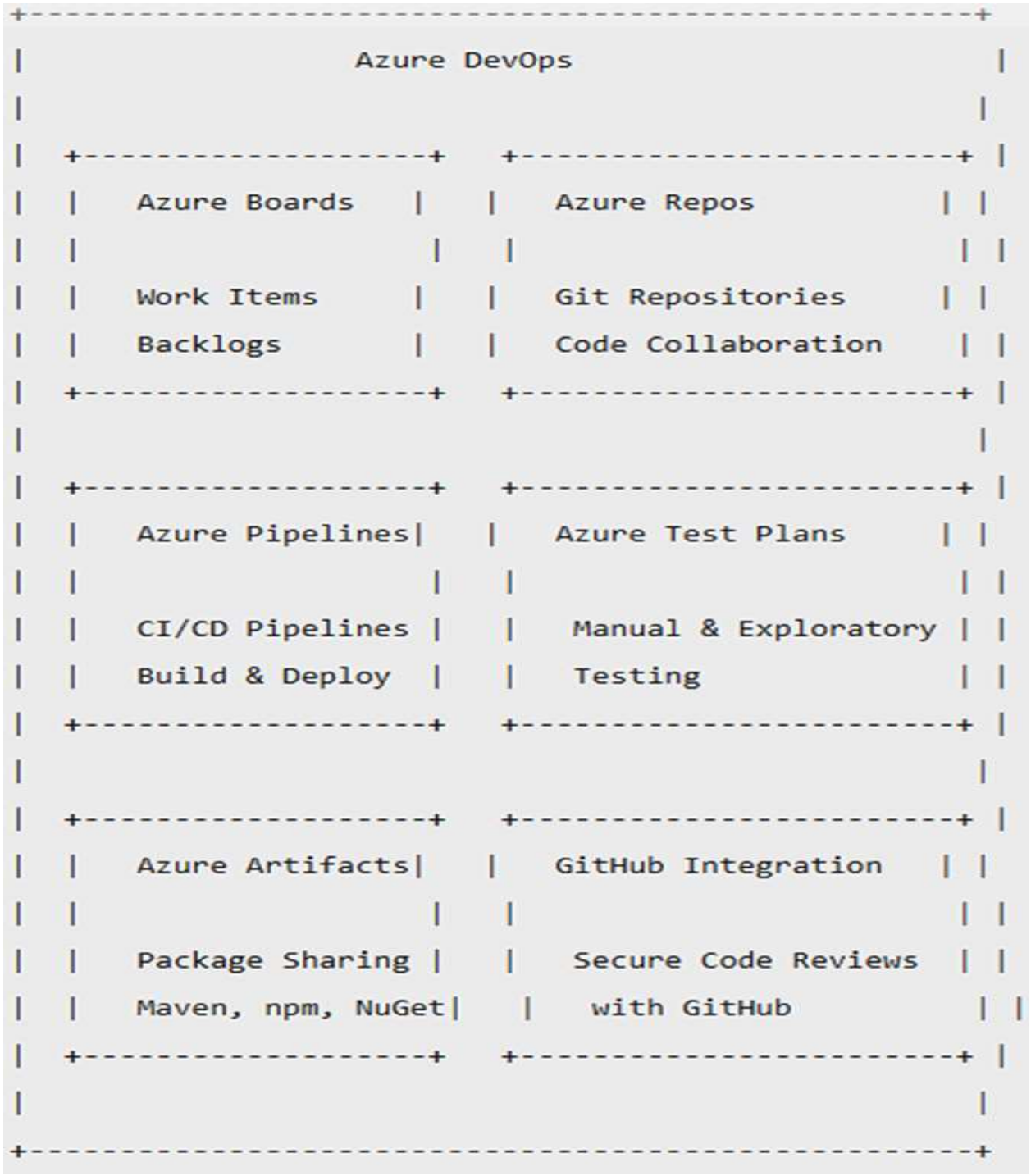
Devops Architecture View

Azure DevOps is a powerful platform for software development, enabling collaboration, automation, and efficient delivery.

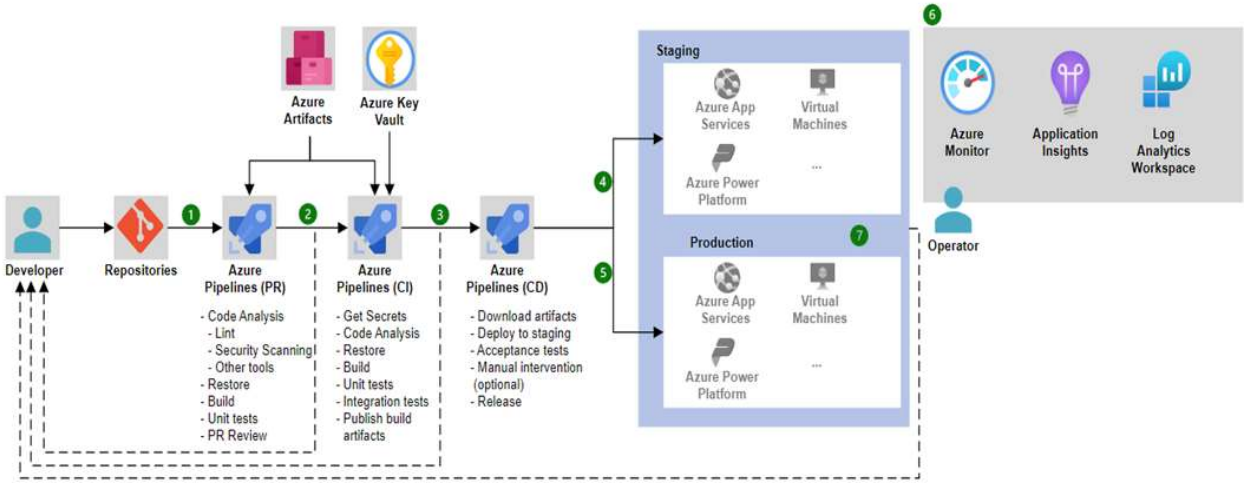
1. **Azure Boards**:
    - Use configurable Kanban boards and interactive backlogs to plan and track work.
    - Achieve unparalleled traceability and reporting for all your ideas, whether big or small1.
2. **Azure Pipelines**:
    - Build, test, and deploy in any language to any cloud or on-premises environment.
    - Run parallel jobs on Linux, macOS, and Windows, and deploy containers to individual hosts or Kubernetes1.
3. **Azure Repos:**
    - Enjoy unlimited free repositories for your code.

- Collaborate effectively with powerful Git hosting and effective code reviews1.

4. **GitHub Integration**:
    - Azure DevOps integrates seamlessly with GitHub.
    - Enhance developer productivity while maintaining security with GitHub Advanced Security1.
5. **Azure Test Plans**:
    - Conduct manual and exploratory testing to improve overall code quality.
    - Release with confidence by ensuring thorough testing1.
6. **Azure Artifacts**:
    - Share Maven, npm, NuGet, and Python packages with your team.
    - Integrate package sharing into your CI/CD pipelines with ease1


Azure DevOps
Azure Boards
Work Items
Backlogs
Azure Repos
Git Repositories
Code Collaboration
Azure Pipelines
CI/CD Pipelines
Build & Deploy
Azure Test Plans
Manual & Exploratory
Testing
Azure Artifacts
Package Sharing
Maven, npm, NuGet
GitHub Integration
Secure Code Reviews
with GitHub

Deployment Architecture of the application



The data flows through the scenario as follows:

1. **PR pipeline** - A pull request (PR) to Azure Repos Git triggers a PR pipeline. This pipeline runs fast quality checks. These checks should include:

    - Building the code, which requires pulling dependencies from a dependency management system.
    - The use of tools to analyse the code, such as static code analysis, linting, and security scanning
    - Unit tests

    If any of the checks fail, the pipeline run ends and the developer will have to make the required changes. If all checks pass, the pipeline should require a PR review. If the PR review fails, the pipeline ends

and the developer will have to make the required changes. If all the checks and PR reviews pass, the PR will successfully merge.

2. **CI pipeline** - A merge to Azure Repos Git triggers a CI pipeline. This pipeline runs the same checks as the PR pipeline with some important additions. The CI pipeline runs integration tests. These integration tests shouldn't require the deployment of the solution, as the build artifacts haven't been created yet. If the integration tests require secrets, the pipeline gets those secrets from Azure Key Vault. If any of the checks fail, the pipeline ends and the developer will have to make the required changes. The result of a successful run of this pipeline is the creation and publishing of build artifacts

3. **CD pipeline trigger** - The publishing of artifacts triggers the CD pipeline.

4. **CD release to staging** - The CD pipeline downloads the build artifacts that are created in the CI pipeline and deploys the solution to a staging environment. The pipeline then runs acceptance tests against the staging environment to validate the deployment. If any acceptance test fails, the pipeline ends and the developer will have to make the required changes. If the tests succeed, a manual validation task can be implemented to require a person or group to validate the deployment and resume the pipeline.

5. **CD release to production** - If the manual intervention is resumed, or there's no manual intervention implemented, the pipeline releases the solution to production. The pipeline should run smoke tests in production to ensure the release is working as expected. If a manual

intervention step results in a cancel, the release fails, or the smoke tests fail, the release is rolled back, the pipeline ends and the developer will have to make the required changes.

6. **Monitoring** - Azure Monitor collects observability data such as logs and metrics so that an operator can analyse health, performance, and usage data. Application Insights collects all application-specific monitoring data, such as traces. Azure Log Analytics is used to store all that data.