Milestone 4: Service Development & Integration

Released: Saturday, Mar 22, 2025

Learning Objectives

- Design and implement a large-scale software system in multi-team settings.
- Design, implement and test an interoperable service to be used by other teams.
- Integrate and test services developed by other teams.

Milestone Tasks

In this milestone, your team will develop a working prototype of the service assigned to your team in Milestone 3, integrate the services developed by other teams, and help other teams integrate your service. This milestone has **two parts with separate deadlines,** where (A) the first 1.5 weeks will be allocated for implementing the shared service that has been assigned to your team and (B) the second week will involve integrating and testing all of the services. **Only part B includes a report submission.**

Part A: Service Development

Due: 11:59 pm Wednesday, **Apr 2**, 2025

Task 1: Service Implementation

Implement the web service assigned to your team. Please ensure that your service conforms to the API you shared and try to not change your API, as the other teams depend on it. In the exceptional case where it is absolutely necessary to modify the API, please notify the course staff and the other teams through Slack. When implementing your service, please carefully consider applying the design principles taught in the lectures, including Design for Change and and Design for Interoperability, to ensure that the implementation is modular, understandable, and compatible with other services.

You may use any available resources for your implementation, including code generation, generative AI, frameworks, libraries, and code snippets from the web. Please document when you use generative AI tools and code from other sources.

Task 2: Service Testing

Develop tests to validate the functionality of your service.

Recall the discussion of **contract testing** from Recitation 6 (March 14). First, develop a set of tests to validate the functionality of your own service. If your service (i.e., **consumer**) depends on another service(s) (i.e., **provider**), create test double components for those services.

Document the expected input-output behaviors of each test double as a **contract**. Publish those contracts to the respective section for the provider service on the shared interface document.

When another team publishes contracts for your service, use those contracts as a basis to create test cases for your own service. Please ensure that your service handles errors gracefully and does not crash on erroneous inputs, as this will make integration easier!

Note: If your team is developing the Central Database Service, you will not need to create any contracts, as your service does not depend on another service. If your team is developing the Data Aggregation Service, you may not receive any contracts, as no other team depends on your service.

Task 3: Service Deployment

After you have implemented and tested your service, deploy the service as an HTTP endpoint on the VM that has been assigned to your team. You will be asked to share the URL and port number for your service with the other teams, so that they can start accessing the service from their own services. To avoid delays in the following integration process, your service must be up and running by the above stated deadline for Part A.

Part B: Service Integration
Due: 11:59 pm Wednesday, **Apr 9**, 2025

Task 4: Service Integration

Replace the test doubles that you used in Part A with the actual services that have been deployed by the other teams. In theory, this step should be as simple as replacing the invocation of a service from a test double to the deployment endpoint for the service. However, there are likely to be unexpected incompatibility issues that arise (possibly due to the changes in the APIs or implicit assumptions about how the other services work). Keep track of these issues as they arise and coordinate with the teams to resolve them. At the end of this milestone, all of the appointment scheduling apps and shared services must be fully functional.

Task 5: End-to-end Integration Testing

The goal of this task is to develop and run additional tests to ensure that your team's implementation (i.e., the scheduling app and the assigned shared service) integrates correctly with the other services. Identify a set of use case scenarios that involve your scheduling app, your shared service, and other teams' services as different components in the overall system. Document these scenarios and develop tests to ensure that the integration works correctly. Sequence diagrams are a good way to document these scenarios, although you do not need to create them for every integration test. You may automate these tests if possible, but a precise sequence of instructions to follow how to run the tests manually is sufficient as well.

Important notes: Since some of the services are shared by multiple teams, please **take extra caution when testing stateful operations**; i.e., those that involve changes to the state of the system (e.g., modify an existing appointment that was created by another team's scheduling app). Doing so may interfere with and cause another team's tests to fail. If you are testing a shared service that involves state changes, make sure that those changes involve test data that was created by your team.

In addition, please avoid overloading the shared services with too many requests at once, as doing so may introduce delays and interfere with another team's usage of the service.

Finally, during the course of the integration process and the future milestones, your team's service may behave unexpectedly or even possibly fail (e.g., when given an unexpected input). Please try to monitor your service from time to time and ensure that it is available throughout the remainder of this project. If, at some point, your service experiences a failure, please try to re-deploy it and make it available to the other teams as soon as possible.

Task 6: Design Reflection

Reflect on your experience implementing, (possibly) re-designing, and integrating your service with the scheduling app and the other teams' services. In particular, discuss the following points:

1. **Design changes**: How did the design of your service or scheduling app change as a result of multi-team integration? Why were those changes necessary? Based on your experience in M4, how would you have designed your service differently in M3?
2. **Communication**: What difficulties did you face in communicating and discussing design decisions with the other teams?
3. **Integration testing:** What challenges did you face in testing the integration of the entire system? How did you overcome these challenges?

Deliverables

Submit a report as a single PDF file to Gradescope that covers the following items in clearly labeled sections (ideally, each section should start on a new page). **Please correctly map the pages in the PDF to the corresponding sections.**

1. **Deployment (1 paragraph)**: Include (1) the URL for accessing the main frontend of your application and (2) the URL for the shared service that your team has implemented.
2. **Service testing report (1 pg max)**: A description of how you tested your own service using test doubles and/or contracts (if applicable to your service). Include links to the test artifacts (e.g., parts of the source code that contains your test cases).
3. **End-to-end Integration testing report (2 pg max):** A description of at least **three use case scenarios** that you developed for testing the integration of your scheduling app, your service, and the other services. Include links to the test artifacts.

4. **Design reflection report** (**2 pg max**): A document reflecting on the process of implementing and integrating your service with the other parts of the system, as described above in Task 6.
5. **Team contract (1 pg max)**: A documentation of (i) the division of development tasks among team members (including the interface person) and (ii) intermediate milestones, each with a target date and goals achieved.

Grading

This assignment is out of **110** points. For full points, we expect:

- **(40 pt)** A functional web service that implements the functionality assigned to your team and a functional appointment scheduling system that integrates with the other services.
- **(10 pt**) A service testing report outlining how you tested your own service.
- **(20 pt)** An integration testing report outlining how you tested the overall system during the integration phase.
- **(30 pt)** A design reflection report.
- **(10 pt)** A team contract that describes (i) the division of tasks among team members and (ii) a set of intermediate milestones, their target date, and expected goals.
- **(5 pt)** Bonus social points (same as the previous milestones).