**Project Intermediate Deliverable Template**

*As a Boilermaker pursuing academic excellence, we pledge to be honest and true in all that we do. Accountable together – We are Purdue.*

*(On group submissions, have each team member type their name).*

Your names: ­­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Today’s date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# **Assignment Goal**

This is a “Deliverable” milestone. You must demonstrate your team’s automated test and deployment pipeline, as well as one or more pieces of functionality as you indicated in your initial plan.

# **Relevant Course Outcomes**

A student who successfully completes this assignment will have demonstrated the ability to

* Outcome i:
  + Identify and follow an appropriate software engineering process for this context.
* Outcome iii:
  + Experience social aspects of software engineering (communication, teamwork).

# **Resources**

One or more intermediate deliverables is a normal component of many software engineering efforts. Such a deliverable gives you the opportunity to demonstrate preliminary functionality to a customer and get feedback. The specific tempo depends on the kind of product, the relationship with the customer, and the team’s degree of experience with “continuous” development processes (Continuous Integration, Continuous Test, Continuous Deployment). At the extreme, some teams demo on a biweekly tempo aligned with their sprints, and even more frequently if they are following the “eXtreme Programming” practice of embedding a customer representative on the team (for obvious logistical reasons, this practice is rare in industry).

# **Assignment**

## Working functionality

Provide prose and screenshot(s) demonstrating all elements of working functionality that were promised by your team in Milestone 1. For example, perhaps you’ve implemented the “*upload package*” feature, and can show a *wget* request or Python client interacting with this feature as deployed on a GCP resource or a local machine.

*(To receive any credit on this part, you must demonstrate at least one working element.)*

|  |
| --- |
| **Completed Feature 1: Prose. (Don’t forget to describe any *planned technical debt*, i.e. aspects of the feature whose implementation you deferred in order to get a working demo. That technical debt should end up in your team’s backlog.)**  **Completed Feature 1: Screenshots**  **Completed Feature 2: …** |

For each incomplete element, explain why you are behind schedule.

|  |
| --- |
| **Incomplete Feature 1: Explanation and progress to date**  **Incomplete Feature 2: …** |

## CI/CD via GitHub Actions

Provide prose and screenshot(s) demonstrating that your team is using GitHub actions to facilitate continuous integration (e.g. by running a *linter* and a *test suite* on every pull request).

|  |
| --- |
| **What steps is your team following prior to accepting a code change? (e.g. git-hooks, code review, linting, test suite, etc.)**  **Provide a link to an example in your GitHub repo where your team followed this CI process (e.g. a pull request):**  **How consistent have you been with this process? What is keeping you from full consistency?**  **What aspects of your system are being tested automatically by your CI scheme?**  **What kinds of defects might go uncaught, and how are you mitigating this risk?**  **Provide screenshots of the GitHub action file (e.g. YAML) that defines the CI stages**  **Provide screenshot(s) of the test suite in action, e.g. the reports from the various tools you have configured, as run on one of your team’s code changes.**  **Describe the extent to which you are able to “continuously deploy”. What is your team’s process to get your current prototype into a deployment on GCP?** |

## Security requirement

The customer requires that you conduct a STRIDE analysis ([link to a notable writeup of this process](https://docs.microsoft.com/en-us/archive/msdn-magazine/2006/november/uncover-security-design-flaws-using-the-stride-approach)). I have advised you to not defer this until the end of the project. Discuss your efforts in this direction so far, and any changes or adaptations you’ve made in your design as a result of your analysis.

## Design changes

Describe any substantial modifications or refinements made to your *design* or your selection of *tools* (e.g. GCP components). Explain why you made any substantial changes.

Did you make changes to the OpenAPI specification provided? What were they, and why did you make those choices?

|  |
| --- |
| **Change 1:** We XXX because YYY.  **Change 2:** …  … |

## Teammate contributions

Fill out this table:

|  |  |  |  |
| --- | --- | --- | --- |
| **Team member** | **Task(s) worked on** | **Task(s) completed** | **Hours spent** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Progress relative to plan

Fill out this table (you can add columns if you want):

|  |  |  |  |
| --- | --- | --- | --- |
| **Task targeted for this week** | **Hours for this task: Planned and Actual** | **Is it complete? (If yes, how did you confirm?)** | **If incomplete: Estimated hours of remaining work** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Are you on, ahead of, or behind schedule? If behind, what is your mitigation plan?

# Grading

This assignment is worth 12.5% of the overall Project grade: 2.5% for the “normal milestone” part, and 10% for the demonstrated functionality (that 10% comes from the 50% total for “Working delivery”).