Office of Strategic National Alien Planning

Development Experience Reporting Platform (DERP)

Product Requirements and Design Document

January 9, 2017

Executive Summary

OSNAP project management requires frequent visibility into developer activity to reduce software development risks. Agile practices, such a daily standups, are costly and inappropriate for the size, geographic distribution, and number of development teams. The OSNAP Development Experience Reporting Platform (DERP) saves thousands of hours of development and project management time by addressing these challenges. DERP also adds new capabilities for continuous development process improvement.

DERP allows developers to connect at anytime from any Internet connected location to support the needs of project and program management. Daily status reports via DERP can be submitted at the developer's convenience and replace the daily standup meeting. Weekly reports are automatically routed to the appropriate project manager for use in executive progress reporting. The reporting records kept by DERP will provide a history for use in project postmortem activities and provide a basis for process improvement. In future revisions, DERP may be integrated with an automated testing framework to allow developers to get early feedback on the integration state of their software deliverables.

Document Versioning

12/28/2016 DE - Initial version

01/03/2017 DE - Updated techstack and software design based on POC

01/09/2017 DE - Rewrite of software design section, add mock pictures

Project Description

Delivery of projects on time with sufficient functionality is a regular problem for development teams. Despite a plethora of project management techniques and practices in industry, the majority of projects remain overbudget, lacking functionality, and late. The OSNAP project management office (PMO) has observed that a major contributor to project failure is a lack of visibility and communication between the development organization and the rest of the business. These communication failures involve insufficient contact, development is highly productive but builds a solution that does not match with the actual requirements, and overwhelming inefficient contact, development is unable to make forward progress due to interruptions and direction changes coming from frequent meetings.

To address these challenges, the PMO has suggested that project management adopt daily standup meetings and weekly contact with the business to verify that development and business needs are tracking together. The process has been effective at increasing project success however the distributed and part time nature of most OSNAP developers has created problems for standup meeting scheduling. Additionally, a large amount of developer time wasted on standup meetings for the larger development teams. Implementation of a software platform to handle the reporting information could save OSNAP thousands of hours per year in time lost to the existing process. The Development Experience Reporting Platform (DERP) will allow OSNAP to realize these savings.

The major cost for the existing process is in the daily standup meetings. DERP's daily report will replace the daily standup meeting. Each day, whenever it is convenient for the developer, the developer will submit a short status via DERP. Each morning at 6am, DERP will send a report of the status messages received for the previous day to the responsible project manager. The project manager will respond directly to individual developers as needed based on the daily status message via email or other mechanism outside of DERP.

Weekly reporting has been an extremely effective method for messaging between the business and development teams on previous projects. Additionally, the history provided by weekly status reporting provides a solid basis for project postmortem meetings. Moving weekly status to DERM will avoid the occasionally lost messages from the current email based process. Like daily status, weekly status reports can be submitted at times convenient for the developers. Weekly status reports will conform to the OSNAP PMO standards and will contain accomplishments, next steps, and upcoming challenges. A weekly status report may also contain general comments. Project managers will be able to view weekly reports through the DERP interface.

User Stories

This section describes several of the user stories that are expected to be covered by the Development Experience Reporting Platform (DERP). User stories are ordered by user type and high level task. Each user story provides a high level user objective and possible user interface process to complete the task.

Developer

User Registration

Objective

Developer would like to register and account with DERP.

Experience

Developer connects to DERP and attempts to login. DERP notices that an account does not yet exist and prompts the user for a DuckId. The user is then presented with the developer dashboard page. Update Profile and Request Testing features are not available until a project manager approves the user account.

Update Profile

Objective

Developer would like to update user/repo information.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects profile. The developer is then able to change their repository URL, preferred email address, and other user editable profile data.

Daily Status Report

Objective

Developer would like to submit a report on progress for the day.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects daily status report. The developer is then able to enter the status message text and submit.

Weekly Status Report

Objective

Developer would like to submit a report on progress for the week.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects weekly status report. The developer is then able to enter the status message text and submit.

Request Testing

Objective

Developer would like to submit their work for early feedback.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects queue integration test. The developer dashboard updates with an estimate of when the integration test will be run.

Status Summary Report

Objective

Developer would like to view daily report history.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects daily status summary. The developer is then shown a summary report giving the count of submitted daily status messages and displaying all the messages in order of submission.

Weekly Summary Report

Objective

Developer would like to view weekly report history.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects weekly status summary. The developer is then shown a list of weekly status report dates, clicking on a date opens the report in read-only.

Code Review

Objective

Developer would like to complete a code review for another developer.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects a pending code review. The developer is then shown the repository URL and a presented with a text area to enter the code review text.

Code Review Review

Objective

Developer would like to view code reviews for the work in their repository.

Experience

Developer connects to DERP and logs in if necessary. From the developer dashboard page, the developer selects view code reviews. The developer is then shown the code reviews submitted about their code base.

Review Testing

Objective

Developer would like to see a integration test results.

Experience

Developer connects to DERP and logs in if necessary. The developer dashboard shows the status of the last test (passed, partial, or failed) and timestamp of test. Clicking on the status opens the text output from the integration test.

Project Manager

User Approval

Objective

Project manager would like to approve a developer.

Experience

Project manager connects to DERP and logs in if necessary. From the manager dashboard page, the manager selects approve waiting accounts. A table listing the GitHub name, DuckId, and when the account was requested is shown. Using a checkbox, the manager selects the accounts to approve then clicks the approve button.

View Daily Status

Objective

Project manager would like to see a summary of daily status reports in DERP.

Experience

Project manager connects to DERP and logs in if necessary. From the manager dashboard page, the manager selects daily status summary and provides a date. DERP shows all of the daily status messages received for that day. The list can be ordered by last name or DuckId.

Receive Daily Status

Objective

Project manager would like to receive a summary of daily status reports.

Experience

At 6am, DERP emails all daily status messages for the previous day. Status messages are ordered by last name.

Review Weekly Status

Objective

Project manager would like to see weekly status reports across users.

Experience

Project manager connects to DERP and logs in if necessary. From the manager dashboard page, the manager selects current weekly status. DERP provides a report showing the previous week's status messages for each user, ordered by last name.

Review User Weeklies

Objective

Project manager would like to see weekly status reports for a single user.

Experience

Project manager connects to DERP and logs in if necessary. From the manager dashboard page, the manager selects view developer history. DERP provides a report showing all the weekly status messages for a selected user, ordered reverse chronological order.

Code Review Approval

Objective

Project manager would like to approve code review text.

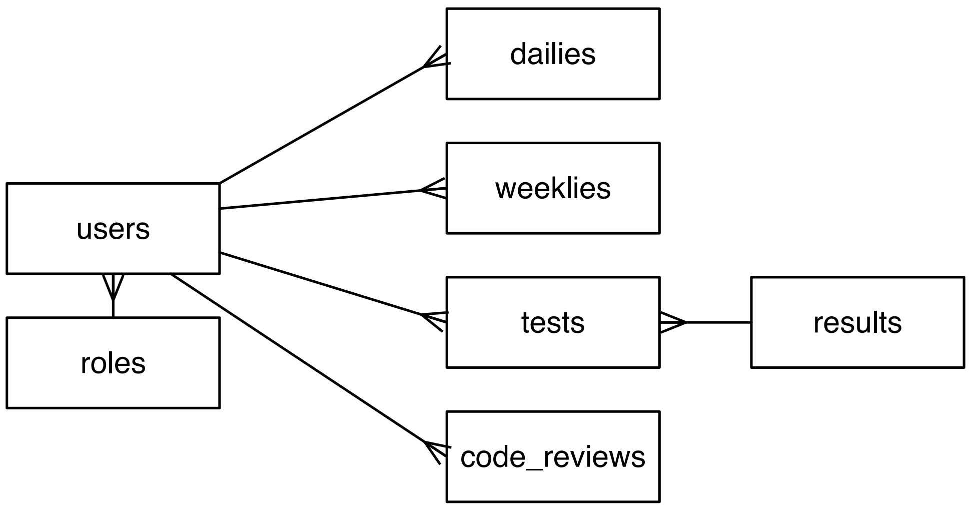
Experience

Project manager connects to DERP and logs in if necessary. From the manager dashboard page, the manager selects approve waiting code reviews. A table listing the reviewer, reviewee, and review text is shown. Using a checkbox, the manager selects the code reviews to approve then clicks the approve button.

Data Model

Persistent data will be stored in a relational database. This section discusses the relational database model supporting DERP.

Overview



Each of the major entities (users, dailies, weeklies, and tests) tracked by DERP have an associated table. Users may belong to a limited set of roles and the roles are enumerated by the roles table. Tests have a current status, the valid statuses are enumerated in the results table.

Table Definitions

users

user_pk	integer	primary key for user records
github_username	varchar(128)	username on GitHub
duck_id	varchar(128)	UO username (to correlate with Canvas)
email	varchar(128)	preferred email address
repo	varchar(256)	https URL for developer's repo
role_fk	integer	foreign key to roles.role_pk

roles

role_pk	integer	primary key for role records
role_name	varchar(128)	string representation

Roles that are expected in the initial version are ‘new user’, ‘developer’, and ‘manager’.

dailies

daily_pk	integer	primary key for daily records
user_fk	integer	foreign key to users.user_pk
create_dt	timestamp	time the daily was created
message	varchar(500)	daily message text

weeklies

weekly_pk	integer	primary key for weekly records
user_fk	integer	foreign key to users.user_pk
create_dt	timestamp	time the weekly was created
accomplishments	text	what was accomplished
next_steps	text	what will be worked on next
challenges	text	anticipated challenges
comments	text	open area for comments

code_reviews

review_pk	integer	primary key for weekly records
reviewer_fk	integer	foreign key to users.user_pk
reviewee_fk	integer	foreign key to users.user_pk
released	boolean	has been released for the reviewee
assigned_dt	timestamp	date the review assignment was made
review_dt	timestamp	date the review was submitted
comments	text	review text

tests

test_pk	integer	primary key for test records
user_fk	integer	foreign key to users.user_pk
create_dt	timestamp	time the test was created
result_fk	integer	foreign key to results.result_pk
complete_dt	timestamp	time the test finished
message	text	test output

results

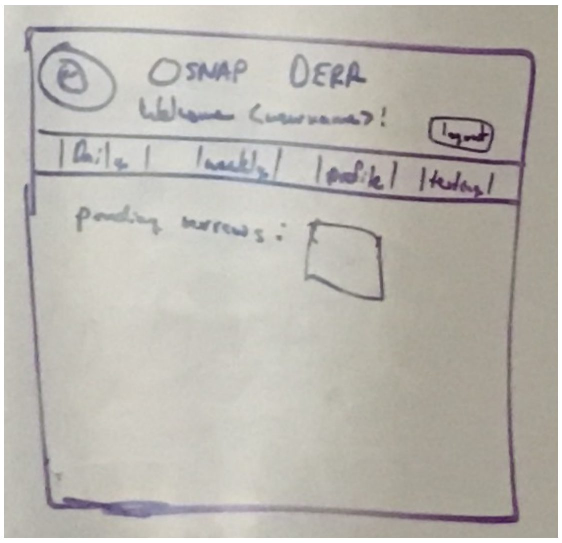
result_pk	integer	primary key for result records
result_name	varchar(128)	string representation

The results table is poorly named. This table enumerates the states a test record may be in. Results that are expected in the initial version are ‘in queue’, ‘cancelled’, ‘running’, ‘passed’, ‘failed’, and ‘partial’.

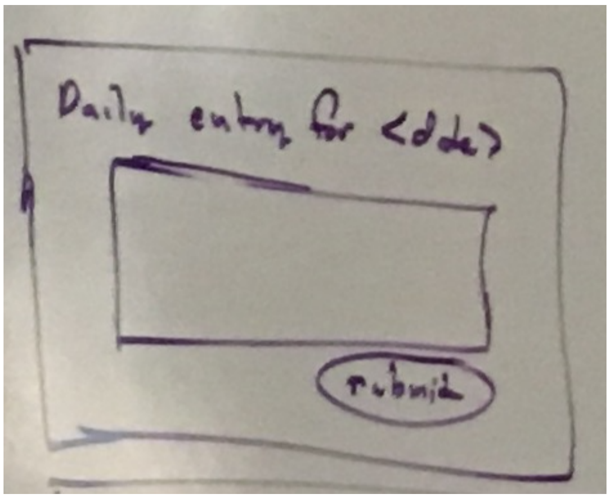
Mockups

This section provides sketches of the expected user interface. Using the screenshots it should be possible to walk through the important user stories and use cases.

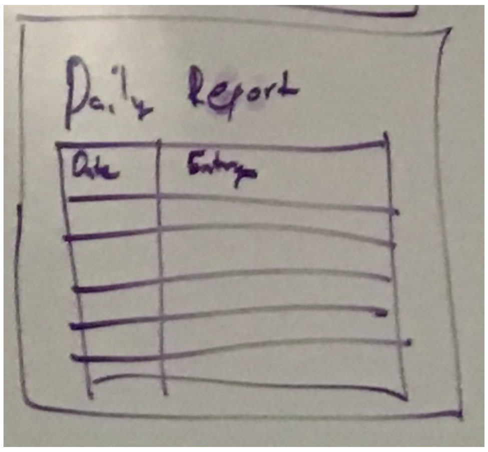
Developer Dashboard



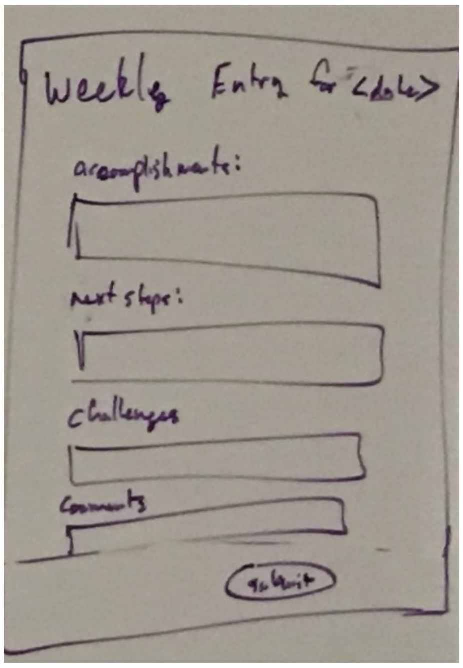
Daily Entry



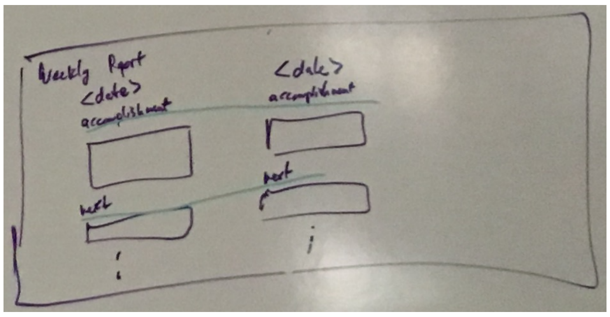
Daily Report



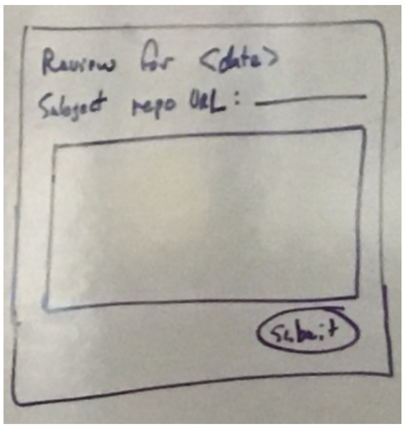
Weekly Entry



Weekly Report



Code Review Entry



Tech Stack

The Development Experience Reporting Platform (DERP) product will use OSNAP's standard web application technology stack. Some additions and deviations to the standard technology stack may be needed to support DERP, these exceptions must be approved by the OSNAP Chief Information Security Officer (CISO) prior to deployment.

Standard Technologies

Apache httpd The Apache http daemon will be used to host the web application.

mod_wsgi mod_wsgi will be used as the gateway between Apache and the application.

Python Python 3 will be used as the application development language.

Flask The Flask framework will support development efforts.

Postgres The Postgres RDBMs will be used for persistent storage.

PGSQL If needed, stored procedures will be written using the default procedure language PGSQL.

psycopg2 The psycopg2 python module will be used by the DERP application code to interface with the supporting Postgres instance.

Approved Exceptional Technologies

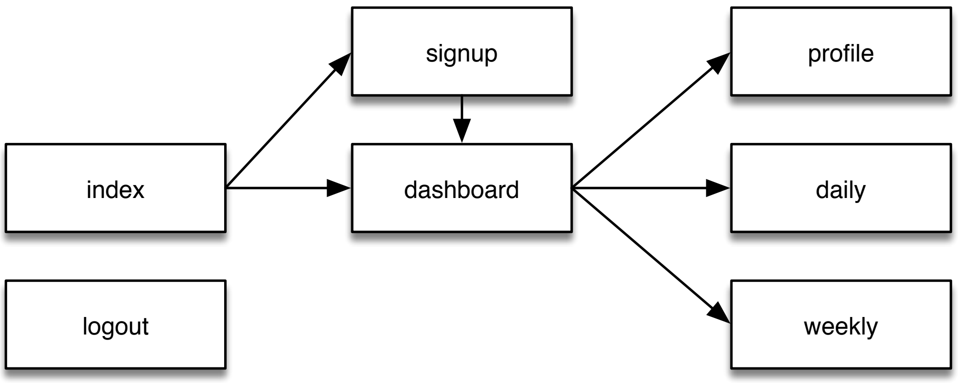
GitHub Oauth GitHub will be used for user authentication. Developers do not necessarily have OSNAP authentication credentials and an alternate service is required.

GitHub-Flask This flask module will be used to handle oauth actions.

Software Design

This section provides architectural details regarding the software design supporting DERP.

Overview



The page flow for DERP is presented in the figure above. The logout page can be accessed by the user selecting to logout or when an error is detected in the user's session. Page names match the names of the implementing functions in *app.py*.

Pages

index

This page is the entry point to DERP and redirects the user based on some checks. If a user must be authenticated, *index* will handle the authentication via GitHub Oauth. When handling authentication, *index* will set the initial values for the user's DERP session.

The user should go from *index* to *signup*, *dashboard*, xor *logout*.

Entry Conditions

Access to this page is unrestricted.

Exit Conditions

signup The user session has a reasonable GitHub credential in the session but no registered Duck Id.

dashboard The user session has a reasonable GitHub credential and matching registered Duck Id.

logout There is a miss match between GitHub credential and Duck Id in the user session.

logout

This page is used to clear the DERP user session. GitHub Oauth uses the GitHub cookie, which DERP cannot touch. Logging out does not change the GitHub user associated with the user's browser.

Entry Conditions

Access to this page is unrestricted.

Exit Conditions

The user session will be cleared.

signup

This page adds a user to the DERP database. The GitHub username is acquired from the session and the Duck Id cannot be changed after submission. Signup also requires a repository URL and preferred email address.

Entry Conditions

1. Valid github_username in the user session.
2. github_username is not present in the database.

Exit Conditions

1. github_username and duck_id are set in the user session.
2. User record has been stored in the database.

dashboard

This page is the user's homepage in the system.

Entry Conditions

1. Valid github_username in the user session.
2. Valid duck_id in the user session.
3. github_username and duck_id match a user record.

Exit Conditions

None

profile

This page allows user settings to be viewed and changed.

Entry Conditions

1. Valid github_username in the user session.
2. Valid duck_id in the user session.
3. github_username and duck_id match a user record.

Exit Conditions

Submitted updates are stored in the database.

daily

This page allows user's to view and add daily status reports.

Entry Conditions

1. Valid github_username in the user session.
2. Valid duck_id in the user session.
3. github_username and duck_id match a user record.

Exit Conditions

Submitted updates are stored in the database.

weekly

This page allows user's to view and add weekly status reports.

Entry Conditions

1. Valid github_username in the user session.
2. Valid duck_id in the user session.
3. github_username and duck_id match a user record.

Exit Conditions

Submitted updates are stored in the database.

Supporting Code

Functionality in this section is organized by the file it appears in.

derp.wsgi

derp.wsgi wraps *app.py* so that mod_wsgi can start DERP correctly.

picklesession.py

Apache wsgi may run several Flask instances concurrently, which is a problem for the default Flask session implementation. *picklesession.py* implements a server side session that is visible across Flask instances on the same host by extending the Flask SessionInterface.

app.py

All of the Flask routes are defined in *app.py*. Additionally, *app.py* contains some helper functions.

is_login_ok(userclass='any')

This function performs the check used by most pages to verify that the user session contains correct information and is authorized to view the page. The following checks must pass for *is_login_ok* to return True.

1. A github_username is in the session.
2. A duck_id is in the session.
3. The github_username and duck_id occur together in the database.
4. The user is of class *userclass*