CS Capstone Design

Technical Demo Grading Sheet (100 pts)

TEAM: SuperGeeks

Overview: The main purpose of the "Technical Demos" is to very clearly communicate the extent to which the team has identified key challenges in the project, and has proven solutions to those challenges. Grading is based on how complete/accurate the list of challenges is, and how convincingly and completely the given demos cover the given challenges.

This template is fleshed out by the team, approved by CS mentor, and brought to demo as a grading sheet.

Risky technical challenges

Based on our requirements acquisition work and current understanding of the problem and envisioned solution, the following are the key technical challenges that we will need to overcome in implementing our solution:

C1: User Interface.

GeekSurvey is a simple web platform to bridge the gap between researchers and participants. As a web application that may have a broad prospects in the future, GeekSurvey should have a good **User Interface** (**UI**) which makes it easy, efficient and user friendly to operate the application. This UI should be mobile friendly, which can be a major challenge for complex applications. It must also be aesthetically pleasing. Poorly designed user interfaces can result in an unpleasant experience, which can be detrimental to the success of the application.

The user interface for GeekSurvey will be created using Django. Django provides a templating engine for dynamically generated web pages. All web pages on GeekSurvey will be some form of a Django template. The demonstration for this challenge will be to create and test a basic "home page" or "landing page" for GeekSurvey. It should have a number of links to other important GeekSurvey web pages, but only the home page has to work for the demonstration. This demonstration should show that Django can be used to efficiently and effectively create a high quality user interface.

C2: Containerization.

GeekSurvey has limited time for development, so it must be packaged as one coherent software product in order to enable future development, configuration, and maintenance. This will be accomplished by including Docker configuration for the web application, so it can be packaged into a single container image (with all dependencies) whenever necessary. Containerization also allows the web application to be very easily distributed and deployed to various platforms including Northern Arizona University Information Technology Systems.

Containerization can complicate development, though. Docker configuration can become difficult for complex applications like GeekSurvey. As of November 2021, the only prototyped GeekSurvey container is an Apache web server, and it only serves a single, basic home page. Ultimately, GeekSurvey will need to run on a container that hosts a complete Django application. This is a totally different challenge from using Apache due to various

interdependencies. The demonstration for this challenge will be a docker container that currently runs and hosts a sample Django application on a specified port. This will show that containerization will be a viable strategy for packaging GeekSurvey.

C3: Tracking Survey Participation.

GeekSurvey must integrate with an external survey platform. It is key to the functionality of GeekSurvey that there is some mechanism to record which GeekSurvey users have completed a given survey. Participants must know which studies they are enrolled in that are in progress or completed. Researchers must know which enrolled participants for a study have completed the associated survey.

Challenges covered by demos:

In this section, we outline the demonstrations we have prepared, and exactly which of the challenge(s) each one of them proves a solution to.

Demonstration 1: GeekSurvey Mock Homepage

Challenges addressed: User Interface

Flight Plan: Step by step overview of demo

1. First, give a short tour of the Django project
2. Then, show the mock homepage
3. Finally, some links from the mock homepage will be clicked

Evaluation:

- ✔ Convincingly demo'd each of listed challenges?
- ✔ Other evaluative comments:

Demonstration 2: Deploy Containerized Django Web App

Challenges addressed: Containerization

Flight Plan: Step by step overview of demo

1. First, give a short tour of the container configuration and files
2. Then, show the Django project files
3. Then, create container image
4. Finally, go to development url to see the web files hosted in our container

Evaluation:

- ✔ Convincingly demo'd each of listed challenges?
- ✔ Other evaluative comments:

Demonstration 3: Mock Survey Workflow

Challenges addressed: Tracking Survey Participation

Flight Plan: Step by step overview of demo

1. First, create a survey using Google Forms
2. Then, show settings available for survey
3. Then, link to GeekSurvey during submission and provide explanation
4. Finally, complete mock survey and demonstrate redirect

Evaluation:

- ✔ Convincingly demo'd each of listed challenges?
- ✔ Other evaluative comments:

Other challenges recognized but not addressed by demo:

If there were challenges you listed earlier that were *not* covered by a demo, list here. This will hopefully be a short list…but better to be clear about where you are. If you have items here, you could list (if applicable) any pending plans to reduce these risks.

1. Payment System

   Plans to Reduce Risk: Research PayPal APIs, specifically PayPal sandboxes and fully deployed environments.