<CODY> About 1:55, then 2:25:

https://youtu.be/sE_1v5ZCQNk

Thank you for joining us for our presentation today of Kashti 2.0. My name is Cody, I'm joined by my teammates Daniel, Derryck, Spencer and Stephen. We would like to show you what we have done this semester for Capstone.

Our objective statement is shown here. Our project is to build a User Interface for an event-driven scripting platform that Microsoft and the open-source community have already created. The product that Microsoft has already created is called "Brigade".

You may be wondering "what is Brigade?" or "what is Kashti?"

Remember that Brigade is an event-driven scripting platform. This platform can be configured to be triggered when any user-defined arbitrary event happens. When the event happens, we can tie that event with a custom script that can be run. Once the script has run, some sort of output can be produced.

That is Brigade. This system can be used for many things, like scripting together multiple tasks and executing them inside containers, batch processing, creating CI/CD pipelines, and some of the examples shown here in the boxes.

Brigade by itself only has a command line interface. Our project is to build a more intuitive, web-based User Interface which is called Kashti.

There is an existing version of Kashti (which is called Kashti 1.0), and you can see one of the pages from it on-screen. Kashti 1.0 was created to work with the first version of Brigade. We will build a 2.0 version of Kashti that is compatible to work with the new version of Brigade that Microsoft has produced.

Kashti 2.0 will focus on redesigning the User Interface to allow Kashti to be used in large-enterprise environments. New features include authentication (you can see there is no login/logout option on Kashti 1.0), interactivity to create, edit or delete items (you can see none of those options exist on Kashti 1.0), and building a TypeScript SDK that interacts with Brigade for Brigade users to integrate it into their web apps.

Those specific features in Kashti 2.0 will be our intended deliverable by summer next year.

<DERRYCK> About 1:37, then 1:56:
https://youtu.be/WV08FnODxlY

Because we're making a user interface for this system, the value that we're adding is not so much architecturally based, but more feature based. So what does this mean? Our work doesn't change the existing architecture. Rather, it focuses more on the features that the user will have access to.

Our project's architecture can be divided up into a few tiers. These of which, are primarily comprised of Kashti, the SDK, and Brigade. As Cody mentioned, Brigade is an event-driven scripting platform for Kubernetes, which runs in a Docker container. For those who may not fully understand scripting, it is essentially a programming language focused on special runtime environments that help to automate tasks. Microsoft has already provided Brigade, which services the back end for our web application. Therefore, our portion of the project will primarily focus on Kashti and the SDK. For Kashti, this is built in Angular (AKA a web framework used for building single-page client applications using HTML and TypeScript), while the SDK is written in typescript. One of the benefits of writing the SDK in typescript, is that it offers the ability to static type. Essentially, it can help the developer catch errors early on in the debugging process. That being said, the SDK will also bridge the user interface as well as Brigade.

Considering that we know an SDK to be a set of tools that allows for the development of applications via a specified platform, it's functionality with respect to this project, primarily involves user project management as well as authentication. As for Kashti, this is the web dashboard for Brigade, which helps the user more easily visualize and inspect certain Brigade events. It also helps provide them with deeper views into Brigade projects, scripts, logs, and jobs. Hence, the whole reason for needing to make a more user-friendly, more intuitive Kashti that will focus on providing the user along with organizations, is the ability to manage and operate projects as they wish or deem necessary.

<STEPHEN> About 1:09, then 1:13:

https://youtu.be/mKcwQ2yh6Bs (only share audio)

Perhaps the primary focus this semester has been writing the Typescript SDK. Brigade does already have an SDK in Go, which provides a bunch of functions to perform authentication, fetch data, etc. However, the current SDK doesn't lend itself to integration with a web app, so part of this project includes a rewrite of this SDK in Typescript. Ultimately it will just hit some endpoints on the Brigade API, so it'll just make it look a bit prettier in the Kashti code and minimize code reuse. An added benefit is that this SDK will make it easier for other developers to build their own applications that interact with Brigade.

One major challenge with this SDK is actually executing the network requests. So far, we've had to rework our network requests twice as we try to zero in on the best method to execute them. However, we do already have a lot of the structure in place and we've already started to add in the specific API endpoints into the Typescript code. So we are making some good progress on it but will still need to work on it towards the start of next semester.

<CODY's YOUTUBE DEMO> 1:07

https://youtu.be/PEfIbbLN9mM

Here is a quick demo of how the TypeScript SDK can be used to access Brigade. We created a simple Node application that imports our TypeScript SDK.

The SDK was downloaded from NPM and included in the dependencies for the Node app.

All that I'm doing with this app is trying to send a GET request to retrieve all the projects that are present in our team's Brigade instance. To do that, users of the SDK need to pass the base URL for their own instance.

The authentication token is passed in the request header and when it is invalid or absent, Brigade will not allow access. The auth token that is currently being passed is empty so we receive an authentication error from Brigade.

But if I populate the auth token to be included in the GET request, Brigade will respond with the data that I requested. See here, we have received a JSON response that includes an array of all the Project items that we have created in our Brigade instance.

This data can be used by Kashti or other apps to easily access Brigade functionality. The TypeScript SDK acted as the bridge between the Node app and Brigade and we hope that Brigade users will also be able to import the SDK for use in their own projects.

<SPENCER> About 1:40:

https://youtu.be/JDY0UHvBPUQ

Although we won't spend too much time talking about security, security is implemented into every step of our development. Kashti talks with Brigade using an activated authentication token which is set by authenticating with Brigade directly using Oauth. This token is stored in local storage. Our two main security concerns here are protecting against session hijacking through exploitation of this stored token, and ensuring that this frontend application (Kashti) is not prone to various attacks.

We are keeping these two items in mind as we review each section of code by comparing our solutions to known best practices and conducting penetration tests. As we develop and review, we are also researching published breaches relating to Angular or TypeScript vulnerabilities.

One example of a breach we read about showed how failure to properly sanitize user input before using that same data to render a dynamic template opened the application up to a XSS vulnerability. Since XSS vulnerabilities could threaten the integrity of our local storage token, we are making sure to specifically secure our code against that said attack.

By implementing strong security review and policies at the start of the project, securing the final product will be a byproduct of consequence instead of a specific additional project that would need to be scheduled on top of our main project's timeline. Although we are encountering the specific challenge of learning to code in TypeScript while also coding securely, we are confident our final solution will be airtight.

<DANIEL>About 52 seconds :
https://youtu.be/QTFgzqYNkyM
In order to visualize our ideas we created a prototype that contains some of the features requested by our Sponsors and enhancements to existing features < click link >This is a high fidelity prototype done in angular. The First feature I would like to show you is login functionality. It is an essential feature for Kashti 2.0 to be authenticated by a third party. The Second feature I would like to highlight is user feedback. It is important to implement in order to inform the user with visual queques on the status of their projects. Another feature we are working to implement is the search function and the ability to add and edit projects. This is still a work in progress but we are confident with the ideas and concepts that we are developing.

<CODY> About 42 seconds, then 45 seconds:

https://youtu.be/kWZjimWilCg

Moving forward, our plan is to complete the project according to the timeline given in this Gantt chart.

Our priority is to complete the SDK so that we can use it as we build the UI and confront implementation issues as soon as possible.

Throughout the process, we will verify security with Quality Assurance practices as we work to ensure that the final deliverable is built on a secure foundation.

After the SDK is complete, we will turn our attention to creating the best possible User Experience in the User Interface. Note that security will be verified as we work on the UI as well as the SDK.

<STEPHEN> About 46 seconds, then 52 seconds:

https://youtu.be/kPY6A_GfF14 (only share audio)

As we finish up the project next semester, we'll also work on deployment. We won't need to deploy it directly since it's just an open source app that the user will deploy, but we do plan on packaging it up to make it easier for them. We'll be particularly focusing on the use case of deploying it through Kubernetes. Since Brigade itself depends on Kubernetes, we can pretty safely assume that anyone deploying this reworked Kashti app will also have Kubernetes in their environment. We'll work on the details of how to package it up for deployment next semester. As we get to that point, we'll also prepare to present our project to the open source community at Kubecon.

Thank you for joining us for our presentation of Kashti 2.0. Now we would like to answer any questions you have for us.