Project Report:

Team 2

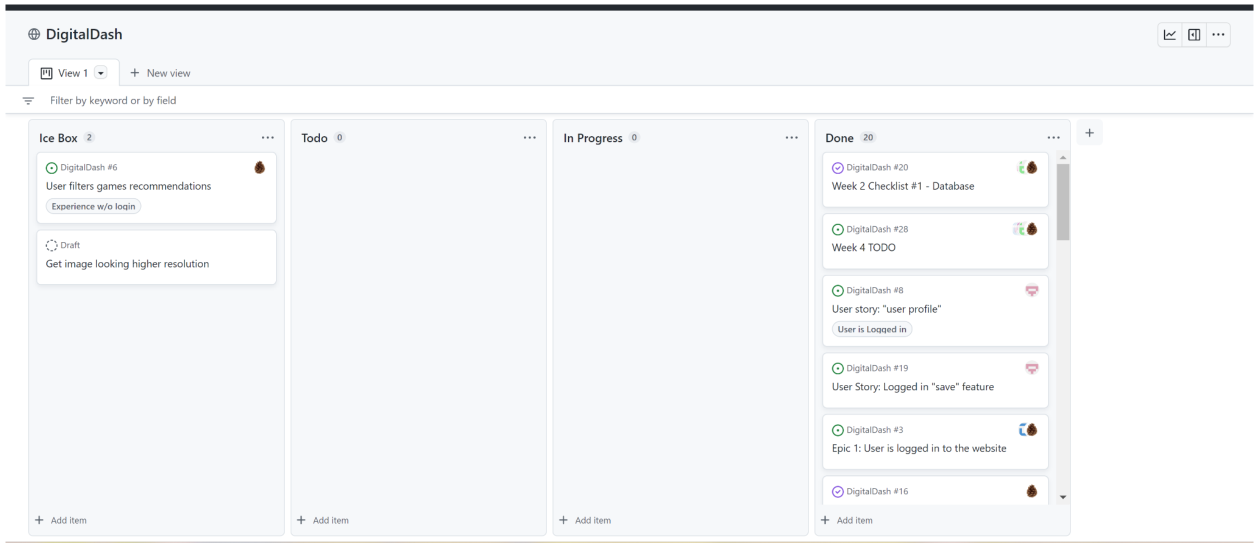
Title: Gameroll

Who: Keith Bates, Kimberly Marthias, Emma Lowery, Kevin McMahon, Joseph Pleasant, and Harrison Pena

Project Description: A 200 word summary of the project

Our main focus for this application was to be a simplistic, random game generator, so that users can spend more time gaming and less time deciding what to play. The application consists of four main pages; a login page, register page, home page and a user profile page. Initially, we planned for the application to open up on just the random list of cards of games that users can click through. Prior to logging in the user will not be able to save or like any of the suggestions. If "save game" is clicked and the user is not logged in then the user will be prompted with the login in page. With-in the user profile page a list of games with the title and a link to more information about the game will be shown. Our main focus is simplicity, and so we went for a modern design with only one game shown at a time. Within the home page, the user will only be able to go to the next game. We created a simplistic application that people interested in gaming can hop on quickly, find a fun, random suggestion to play next and hop off to go try a new game.

Project Tracker - GitHub project board:

- https://github.com/users/dnerever/projects/1
- Screenshot showing your project in your project tracker
- 

DigitalDash

View 1 · New view

Filter by keyword or by field

Ice Box 2
- DigitalDash #6 — User filters games recommendations — Experience w/o login
- Draft — Get image looking higher resolution

Todo 0

In Progress 0

Done 20
- DigitalDash #20 — Week 2 Checklist #1 - Database
- DigitalDash #28 — Week 4 TODO
- DigitalDash #8 — User story: "user profile" — User is Logged in
- DigitalDash #19 — User Story: Logged in "save" feature
- DigitalDash #3 — Epic 1: User is logged in to the website
- DigitalDash #16

Add item

Video: https://www.youtube.com/watch?v=eBcX9KZa30c

VCS: Link to your git Repository. Instructor/TAs will check, weekly, to ensure the following are stored in your VCS repository:

- Source Code
- Test Cases
- Video demo
- README.md in GitHub
- Project documentation
- Project Board

https://github.com/dnerever/DigitalDash

Contributions:

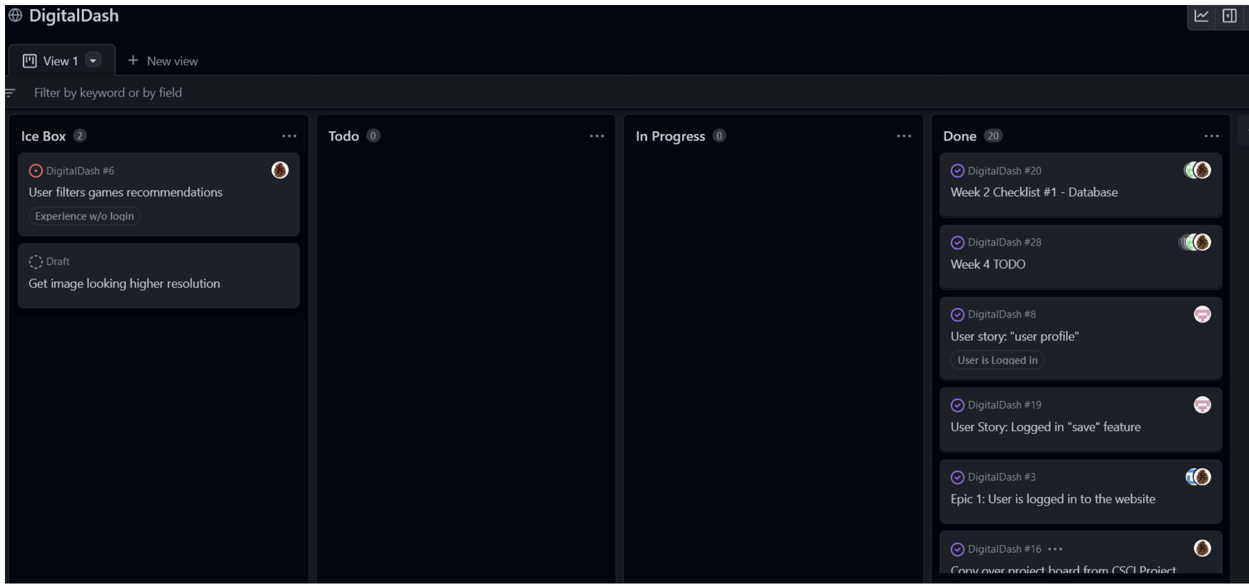
- A brief (not more than 100 words) from each team member about their contributions.
    - This should include the technologies worked on
    - Features that have contributed to
    - Keith Bates: Starting off I worked on finding a usable API or dataset and also handled a majority of git issues with branching and merging. I also worked on building the login, logout .ejs pages with Kimberly. Then I worked with Emma and Kevin on getting our API connection working using postman and NodeJS. During the final couple weeks I spent my time implementing randomization from our API data, working together on our /savegame and /nextgame NodeJS routes, troubleshooting our db inserts and queries and a lot of general troubleshooting across all parts of our application.
    - Kimberly Marthias: In the beginning steps of the development of the project, I contributed to the function of the login and logout page and made sure that those were functioning correctly including the saving of user information to the database and password hashing. I also contributed to the functionality of the home page. In the final stages of development, I contributed to the styling of the homepage by changing background color and font.
    - Emma Lowery: In the beginning of the project I helped set up the foundation code of our website including the navigation menu, header/footer files and the outline of our index.js file. My main focus over the course of the project was the homepage. I assisted in designing the layout in the beginning stages and focused more on the backend API calls towards the end of the project. Focusing heavily on how to pull data from the API and display it on our webpage. I also worked a little bit on the security between our API and our own database.
    - Kevin McMahon: In the early stages of the project, I worked on the route of the register page. After completing that relatively simple task, I helped design the layout of our database, and how we should be organizing data so our website

could run as efficiently as possible. Towards the end of our project, I created the save game route, and fully implemented the delete saved games feature. Additionally, in the later weeks, I created the release notes and updated the project board before we met with Cory for our weekly meeting.

- Joseph Pleasant: For my contributions, I helped build the register page, and primarily worked on the profile page. Using the api data called to our home page, I worked on building the UI of the profile page. Making the cards for the saved games and creating the format of how the information is displayed on the card. I also added styles for our website's UI. I incorporated a SNES open source font library to give our site a signature video game feel. I also wrote the routes for the profile page and controlling our authentication system to make sure users register or sign in to use certain features on our website.
- Harrison Pena: The home page is where I started, I worked with Emma and we diagramed a very simple layout for the website. For any design in this project I usually pushed to reconsider any unnecessary buttons or info with the hope of making a page simpler and easier to use. I reordered and resized the home page many times along with my group until we reached a point where it felt right. For the second half of our time I worked on the Save Game button and getting that to correctly call and add to the user database of saved game info.

You can also include:

- A screenshot of the project Board



DigitalDash

View 1 | + New view

Filter by keyword or by field

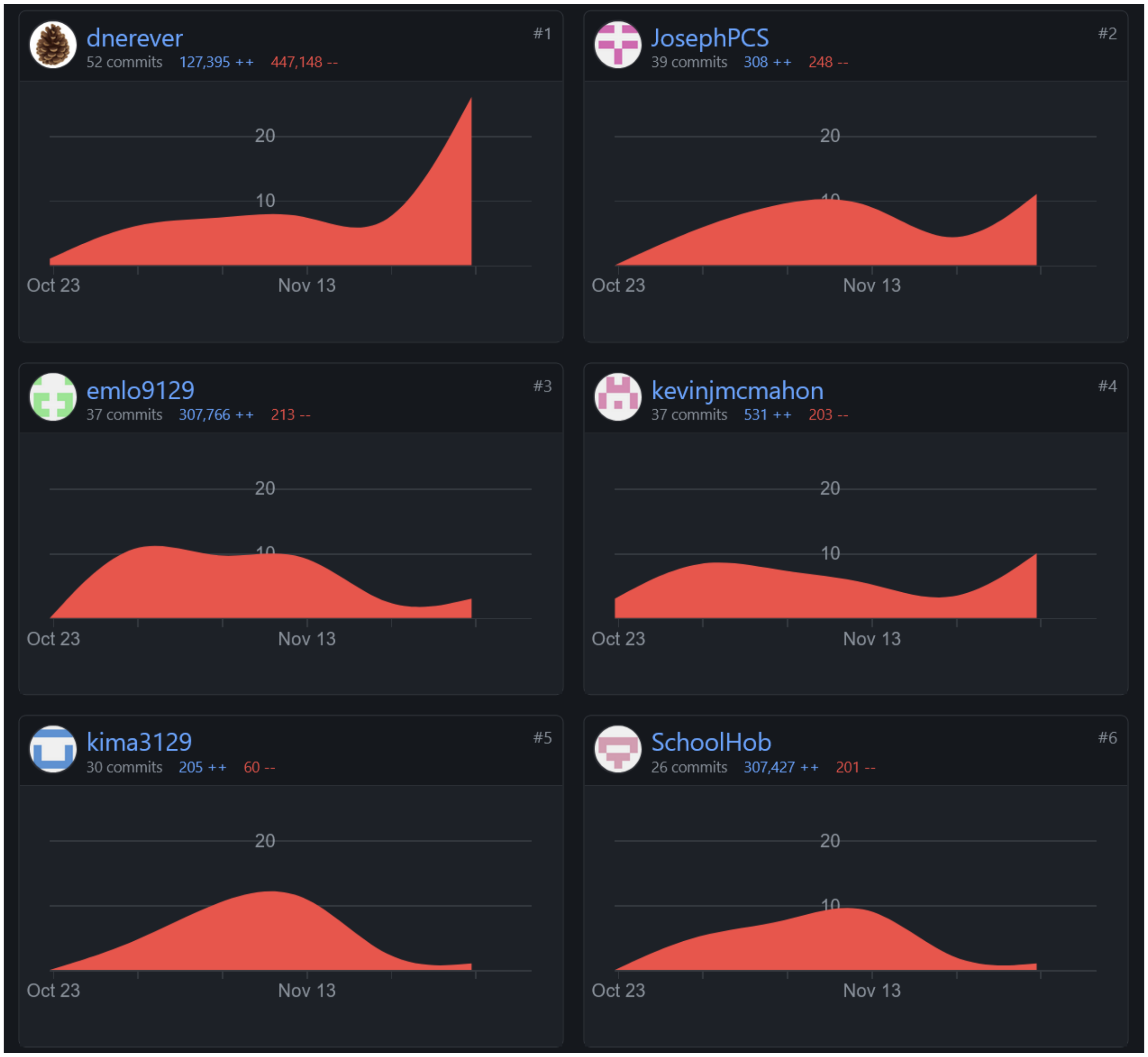
Ice Box 2
- DigitalDash #6 — User filters games recommendations — Experience w/o login
- Draft — Get image looking higher resolution

Todo 0

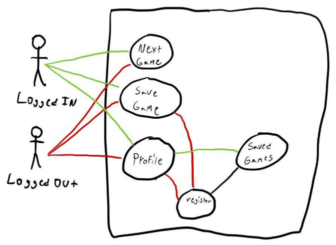
In Progress 0

Done 20
- DigitalDash #20 — Week 2 Checklist #1 - Database
- DigitalDash #28 — Week 4 TODO
- DigitalDash #8 — User story: "user profile" — User is Logged in
- DigitalDash #19 — User Story: Logged in "save" feature
- DigitalDash #3 — Epic 1: User is logged in to the website
- DigitalDash #16 — Copy over project board from CSCI Project

- A screenshot of the contributions on GitHub:



Use Case Diagram: You need to include a use case diagram for your project. You can build on the use case diagram you created in the proposal. If you built a complete use case diagram for the proposal, you can include it as is.



Test results: In Lab 11, you created a Test Plan. You need to include the test results and observations in the project report. Refer to this for more information

1. **Feature one: Testing game randomization**
    a. User intuitively knew how to get to the next game and declared games to be random (20 games cycled; no repeats)
2. **Feature two: Test if it redirects to registration page if clicked on the like/save button and user is not logged in**
    a. User attempted to save while not logged in but was prompted by the website to register or login and proceeded to register and then login.
3. **Feature three: Test if saving a game properly save into your profile and stay there**
    a. Once logged in user decided to save a game again without any instruction from dev. Also user discovered how to remove game from saved games after accidentally saving a game.

User successfully navigated the site and was able to intuit all features, also the site ran perfectly and all data was saved in the user even when logged out and back in. One note they had was that the “New Game >>>” button could be changed to “Next Game >>>” to avoid any confusion.

Deployment Methods:

1. Steps to deploy Gameroll on your CU private server
    a. Change the docker-compose.yaml file web port from ‘3000:3000’ to ‘:3000’ and push your change to the repo.
    b. While connected to the campus internet, use ssh through your terminal of choice to connect to your private server. The command is:
    ssh <YOUR_IDENTIKEY>@csci3308.int.colorado.edu

c. Once the connection is established navigate to the directory you would like to have the deployment in.
d. Install rootless-docker with the command:
   dockerd-rootless-setuptool.sh install
e. Now clone the DigitalDash repo with the command:
   git clone git@github.com:dnerever/DigitalDash.git
f. Change the permissions for your repo with two commands:
   chmod 777 .
   chmod 766 package-lock.json
g. Now you will need to add a .env within the gameroll/ directory
   i. You will now need to create a Twitch API key which can be done by following IGDB's guide (They explain it better than I could :))
   ii. Also here is IGDB's guide that you must follow to get your bearer access token GUIDE
   iii. Navigate to gameroll/ with command:
      cd gameroll/
   iv. Now create a .env file:
      touch .env
   v. Now that we have both our Client ID and Bearer Token we can enter those in our .env file using vim with the command:
      vim .env
   vi. Vim file example:
```
# database credentials
POSTGRES_USER="postgres"
POSTGRES_PASSWORD="pwd"
POSTGRES_DB="gameroll_db"

SESSION_SECRET="super duper secret! Gameroll"
client_id="<YOUR_CLIENT_ID>"
authorization="Bearer <ACCESS_TOKEN>"
```
   vii. Now save your .env and close the vim editor
h. Ensure you're in gameroll/ and run:
   docker-compose up -d
i. Then to find the exposed ports run:
   docker-compose ps
j. Now the website can be accessed through a browser at the url:
   http://csci3308.int.colorado.edu:<YOUR_EXPOSED_DOCKER_WEB_PORT>
   (In our case it will be accessible through: http://csci3308.int.colorado.edu:3000)
k. That's all! Enjoy all of your new games (We do have a hosted version of our application)

2. Steps to deploy locally
    a. Make sure you have docker compose installed
    b. In a terminal of your choice navigate to your desired location for the repo
    c. Once there clone the repo:
       git clone git@github.com:dnerever/DigitalDash.git
    d. Next navigate to the gameroll/ folder:
       cd gameroll/
    e. Now you will need to add a .env within the gameroll/ directory
        i. You will now need to create a Twitch API key which can be done by following IGDB's guide (They explain it better than I could :))
        ii. Also here is IGDB's guide that you must follow to get your bearer access token GUIDE
        iii. Navigate to gameroll/ with command:
             cd gameroll/
        iv. Now create a .env file:
            touch .env
        v. Now that we have both our Client ID and Bearer Token we can enter those in our .env file using vim with the command:
           vim .env
        vi. Vim file example:
```
# database credentials
POSTGRES_USER="postgres"
POSTGRES_PASSWORD="pwd"
POSTGRES_DB="gameroll_db"

SESSION_SECRET="super duper secret! Gameroll"
client_id="<YOUR_CLIENT_ID>"
authorization="Bearer <ACCESS_TOKEN>"
```
        vii. Now save your .env and close the vim editor
    f. Ensure you're in gameroll/ and close all other docker containers with:
       docker-compose down –volumes
    g. Next run:
       docker-compose up -d
    h. Wait about 5 seconds until the API connection is made
    i. Now you can open a browser and go to the url:
       localhost:3000
    j. All Done!