FURRY-MINDER

Team Members: Sarah Jorissen, Laura Lopez, Jonathon Hoffman

What will be covered

We will briefly discuss:

1. What Furry-Minder is
2. The technology used throughout development
3. The design of the application
4. The deployment of the application
5. Future work

What is Furry-Minder?

Furry-Minder helps to incentivize the user to stay productive and on task by tying the care of the pet to a to-do list. If tasks are finished in a timely manner, the pet will be fed & cared for, and continue to grow. However, neglecting your tasks also means neglecting your pet, which will eventually lead to its death, and starting all over again.

Technology

Browser: The primary browser that Furry-Minder was developed for was Google Chrome.

- We decided that Google Chrome was the best choice because everyone on the team was most familiar with it.

OS: The operating systems used during development were Windows and Ubuntu.

- The main operating system we focused on was Windows due to the majority of the team using Windows for development.

IDE: The IDE that we chose was WebStorm IDE by JetBrains.

- We chose this IDE because it was a quality JavaScript IDE that works on multiple operating systems and helps simplify and streamline many necessary tasks. Free to use for college students so all team members can access it.

Technology Cont.

Languages: We decided on using HTML, CSS, JavaScript, and TypeScript

- ❖ We chose these languages because we are developing a single-page web-based application

Library: We decided on using the React library to assist with the development of the application.

- ❖ We chose React because the library helps streamline the creation of a React project and comes with various useful features built-in. It will help ease the process of starting and working on a React app for a team that is mostly new to React.

API: For the calendar component of the project we decided on using the FullCalender api

- ❖ We chose this because the FullCalendar api provided all features we really needed and allowed us to focus more on other parts of the project.

Database: For the database, we chose Firebase.

- ❖ We chose Firebase because it seemed like a really good fit for the project with more than enough documentation.

Technology Cont.

Communication: For group communication we decided on mainly using Outlook and Discord.

- ❖ We decided on Outlook and Discord, because the team was most comfortable with these applications.

Project Management: To assist with managing tasks and to keep all members on the same page we used a to-do list also included in our Discord to maximize communication of tasks that were needing to be done.

Version Control: For version control of the project we used GitHub per request from the Professor.

- ❖ This decision was made for us, but we have all noticed that it was detrimental to the success of the project.

Design

In this section we will discuss a brief overview the design of the project.

We had 4 main components:

- The Pet Component
- The Task Component
- The Calendar Component
- The API Class

Calendar Component

- The Calendar Component has several features that allows the user to traverse the dates of the year.
- The Calendar provides three different views
  - Weekly (default)
  - Monthly
  - A list view
- The Calendar uses state and dependencies to maintain correctness of tasks being displayed.
- Calendar component also allows users to edit tasks that are displayed on calendar

Task Component

- Allows users to add tasks onto the task list using the name, description and due date
- Tasks will be displayed for the current date on the task list
- Users are able to mark their task as completed via checkbox
- User will be able to view details of their task by hovering over each task
- Users will be able to edit their submitted tasks and they will be updated in the database and live screen
- Users will be able to delete tasks

Pet Component

- ➢ After registering an account, the user will be prompted to give their pet a name, and one will be randomly generated and displayed next to their task list
- ➢ The pet starts out as a child, and will grow into its ‘teen’ phase when it reaches 3 days old, and finally its ‘adult’ phase after a week
  - ○ As it grows, the pet’s HP will increase, starting from 50 HP up to 100 HP
- ➢ If the user does not complete tasks, the pet’s HP will be reduced for each one that isn’t complete
- ➢ If the pet’s HP reaches 0, it will die, and the user will be prompted to create a new pet, which will once again start off as a child

API Class

- API class consists of several methods that allows the other components to access Firebase.
  - getPet() method
  - createPet() method
  - updatePet() method
  - addTask() method
  - getTask() method
  - getTaskByDate() method
  - editTask() method
  - deleteTask() method

Deployment

- Firebase provides emulators for authentication, hosting, and the database that serve as a replacement for their production counterparts during development
- When the project was ready to deploy, we ran the build command to bundle the project and allow it to be run on the browser
- When a user visits the site, the entire application is downloaded and executed completely in their browser and interacts directly with the database

Future Work

- Toggle between dates in the task list component
- Ability to set tasks to be repeated
- Add more species and color options for pet
- Add the ability to customize pet with accessories
- Stricter password requirements

Summary

- What Furry-Minder is
- The technology used throughout development
- The design of the application
- The deployment of the application
- Future work

Closing

FURRY-MINDER

From all of us at Furry-Minder …. Thank You!