# CS 340 README

## About the Project/Project Title

*The README file for Zircon, a user-friendly tile engine and text GUI, explains its purpose as an extensible tool for creating tile-based games and applications. It highlights features like multiplatform support, built-in fonts, tile sets, and custom tile set format. The Bigfoot Sightings Dash App, built with Plotly's Dash framework, explores Bigfoot sightings data with various plots, a grid layout, input field interactions, and caching. The Dash framework, built on Plotly.js, React, and Flask, is a Python framework for building ML and data science web apps. It ties modern UI elements to analytical Python code and provides examples of Dash apps.*

*The README for Zircon, a user-friendly tile engine and text GUI, provides an overview of its features, including drawing, input handling, layering, animations, shape and box drawing, fonts, and tile sets. It also lists necessary tools for dependency management and provides instructions on adding Zircon as a dependency. The README for the Bigfoot Sightings Dash App explains its use with Plotly's Dash framework, focusing on exploring Bigfoot sightings data. It includes features like plots, grid layout, interactions with input fields, and caching. The README lists tools and libraries used, including Python, Plotly's Dash, and Bootstrap2, and provides step-by-step instructions on setting up the environment, installing dependencies, and running the app locally. The README for Dash, a Python framework for building ML and data science web apps, highlights its capabilities and provides examples of apps.*

Motivation

Zircon is a user-friendly tile engine and text GUI designed to simplify game development by offering an extensible and multiplatform tool. It aims to address common challenges in game development, allowing developers to focus on creativity rather than technical complexities. The Bigfoot Sightings Dash App uses Plotly's Dash framework to explore and visualize Bigfoot sightings data, making data analysis more engaging and accessible. Dash bridges the gap between Python code and modern web-based user interfaces, enabling data scientists and analysts to create interactive, web-based applications without extensive web development skills. It leverages Plotly.js, React, and Flask to create powerful frameworks for building machine learning and data science web apps.

## Getting Started

To begin using Zircon, a user-friendly tile engine and text GUI, follow these steps: clone the repository, install dependencies, run the application, and explore the documentation. For Bigfoot Sightings Dash App, clone the repository, create a virtual environment, install dependencies, run the application, and access the app.  
 To get started with Dash, install it, create a new Dash App, and add the following code:  
import dash, import dash\_core\_components as dcc, and import dash\_html\_components as html. Create a new Python file (e.g., app.py) and add the following code:  
app = dash. Dash(\_\_name\_\_)

app.layout = html. Div(children=['Hello Dash'

## Installation

Zircon is a user-friendly tile engine and text GUI that requires Java Development Kit 8 or higher, either from the Oracle website or an open-source alternative like Adopt OpenJDK. It supports both Maven and Gradle build tools, and Git is required to clone the repository. To install Zircon, git clone https://github.com/Hexworks/zircon.git and cd zircon. Install dependencies using mvn install or Gradle build, and run the application using mvn exec: java -Dexec.mainClass="com.yourpackage.Main" for Maven and Gradle run for Gradle. To use Dash, Python 3.6 or higher, create a new Dash App, and install it using pip.

## Usage

*Use this space to show useful examples of how your project works and how it can be used. Be sure to include examples of your code, tests, and screenshots.*

### Code Example

*The code enables users to test, add, and edit animals in a shelter using mongo and Python files. To add animals, users can enter "animals.create (STRING\_TYPE)" and the program will throw a Boolean if successful or an error if not added.*

### Tests

*The code was tested using an invalid statement, creating an invalid data type. To ensure the added animal was added, use query = animals.read({"name": "NAME”).*

### Screenshots

*A screenshot of a computer

Description automatically generated*

## Contact

Casey Farmer: