launch_code

LC101 2.12

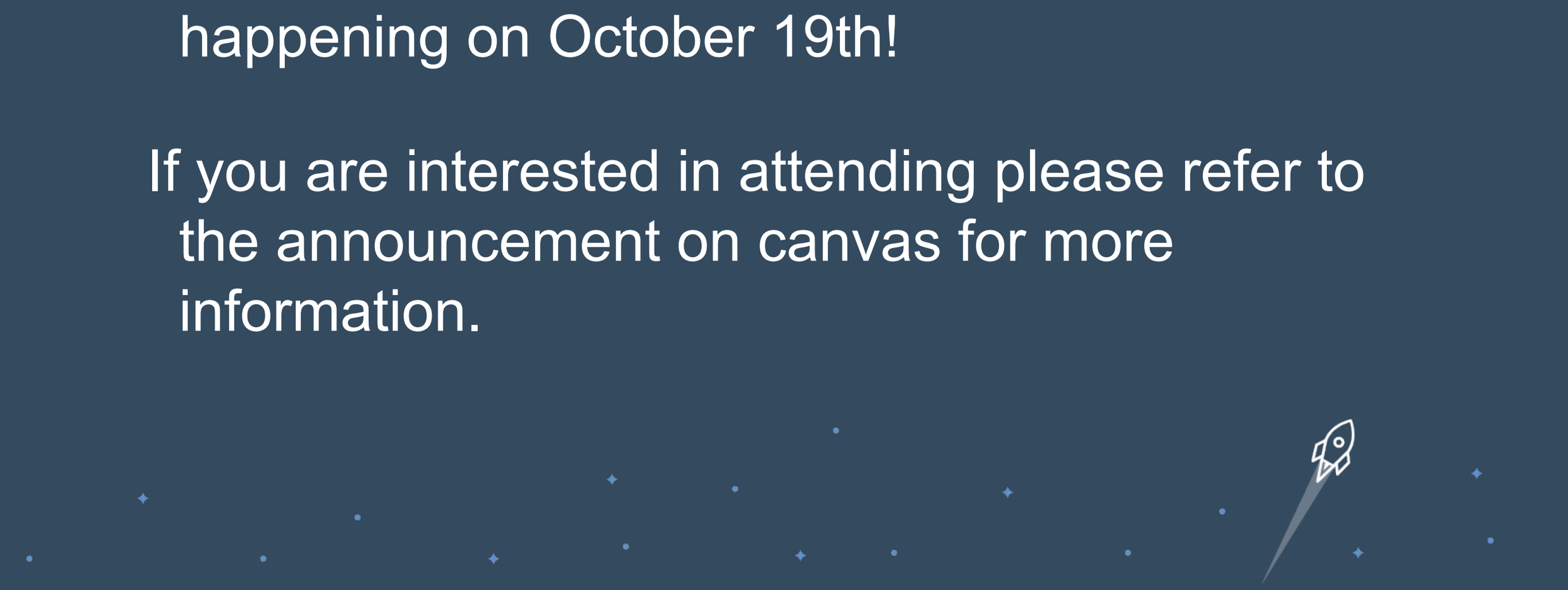
Class Agenda

1. Announcement (Geek Gala)
2. Last Class Review
3. Lecture
4. TF Work groups!

Announcement

GeekGala - LaunchCode's birthday party is happening on October 19th!

If you are interested in attending please refer to the announcement on canvas for more information.



Announcement 2

We have a guest speaker after the lecture. I will introduce him when he arrives.

Review

Cookies - temporary storage on the client's browser.

Cookies are set by the HTTP response, and set via the client's requests.

Cookies can be disabled by users, and so aren't always a reliable way of storing information, but are still very handy tools for optimization.

MVC

An architectural paradigm.

It defines how our application fits together. It separates the sections of our application (providing organization, and reducing the total code we need to write), and provides a guide for future developers.

MVC

Model - The data! In an application it's the ORM, and the classes. It's also the entire Database.

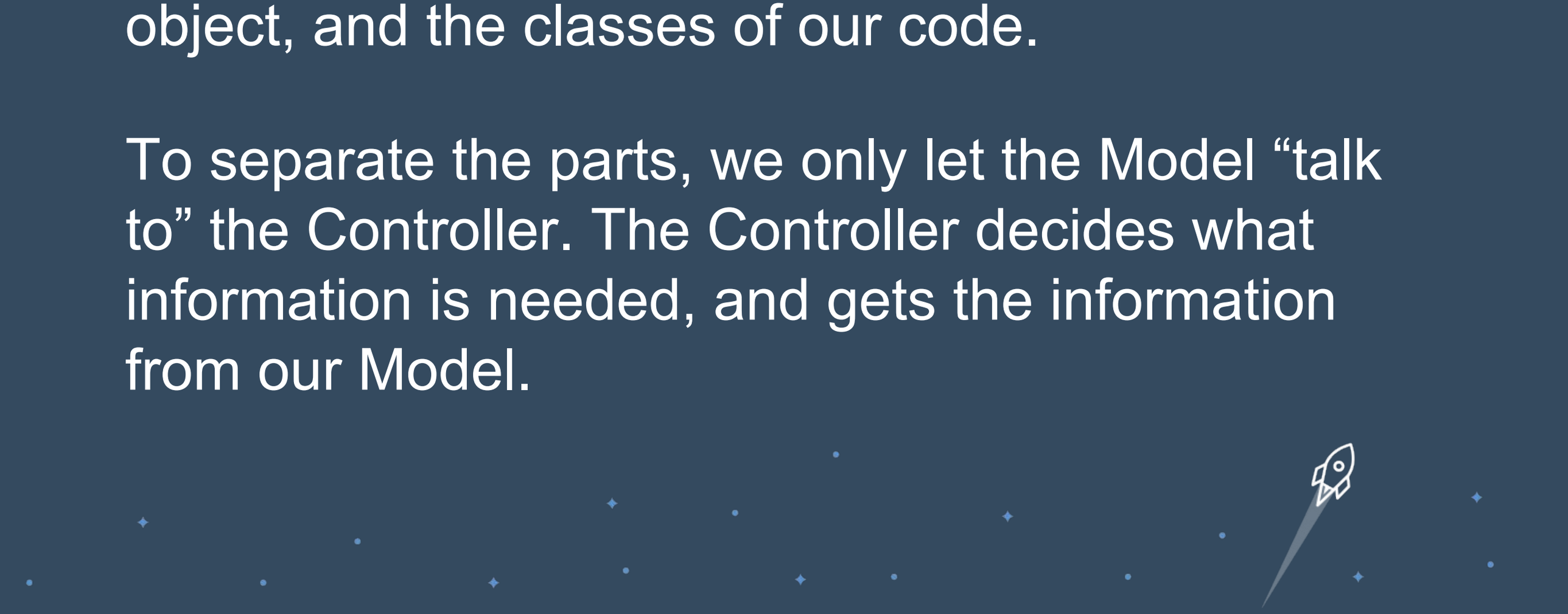
View - What the user sees, and interacts with! HTML/CSS/JavaScript, usually dynamically built with a template engine.

Controller - The logic! Route decorators, and functions that define what happens when a user requests a webpage, or click a button. It handles our methods (GET, POST)

Model

MySQL Database, SQLAlchemy (our ORM), the db object, and the classes of our code.

To separate the parts, we only let the Model “talk to” the Controller. The Controller decides what information is needed, and gets the information from our Model.



View

Our templates, our CSS, and JavaScript make up our View.

All the View does is display/gather information from the User, and passes it to the Controller.

Controller

@app.route()

@before_request()

All of the functions we have built after our decorators. They take the input from the forms (in our View), and pass/request that information to our DB (the Model)

The Controller is the middle man, it lives between the View, and the Model and contains the logic that determines what is put into, or taken out of the DB.

Why are we learning this?

MVC, and architectural patterns in general, are how companies structure and organize their projects.

These patterns allow them to separate the various functions of their projects allowing them to assign different parts to different developers. And they can more easily plan, and code for different devices (laptops, phones, tablets).

In Unit 3, Java (and C#) use MVC patterns to structure their codebases, and we will expect you to understand the three pieces.

Studio Flicklist 8

Flicklist 8

Then catch up on any assignments you are still working on.