**Integrate and Configure:**

Utilizing the selected components, the system will need to be designed with them in mind.

Given the requirements of the chosen existing software modifications will need to be made to the system

Researching existing tools could help tremendously to make development easier. As an example, ORMs provide an easier way to interact with databases.

Create a scheduling system like leopardWeb. The system should utilize a database of users, and should implement different user roles including instructors, admins, and students.

System validation can be handed by both human users and through unittests to check basic functionality like updating users and classes.

System validation can be handed by both human users and through unittests to check basic functionality like updating users and classes.

Requirements Specification:

* Create a scheduling system like leopard web
* Utilize databases to store user information
* Create an easy-to-use interface for users
* Provide different levels of user access depending on the classification of the user. For example, students should not be able to add or remove classes from the system

Component Analysis:

* ORMs like Peewee allow databases to be easily controlled in Python.
* Web frameworks like Flask and Django allow web interfaces to be created in Python

Requirements Modification:

* When designing the database control node keep in mind that peewee should be used. Additionally, when interfacing with the database use Peewee.
* When designing nodes make sure that data is returned rather than printed from every function so that it is easy to add to a web interface

System Design with Reuse:

* Use Peewee in the database control node, utilizing a single node to handle all database transactions will prevent a querying the database simultaneously which is not supported by SQL Lite
* Utilize flask to create a web interface, this interface can be designed like any other website using a combination of HTML templates and Python.

Development and Integration:

* Over the course of the semester work on the application and add all functionality specified above
* Make updates when additional functionality is needed

System Validation:

* Utilize python unittest framework to create tests for every function
* Have human users test the program and attempt to find bugs
* As user feedback is returned from end users make improvements as necessary.