Final Project Description:
The main intent of the final project is to exercise the skills required of an iOS application developer, as learned in the class lecture and through auxiliary reading. In order to reach this goal, the project will need to meet a set of required criteria and also include a minimum of two optional criteria as outlined below.

To ensure that progress is being made on the project, and to emulate working on a project for a real client, various milestones will be assigned. At these milestone dates the project will be reviewed to see that it meets specific requirements for the milestone as described below, or in the case of the first milestone, a project proposal will be due.

For the duration of the project a personal git repository that must be used to support final project development efforts will be provided. This repository will also be utilized by the instructor to retrieve projects for evaluation (either at due dates or for help at the request of a student). These repositories will be deleted at the conclusion of the class, so keeping a local clone of the repository for continued work is suggested.

Final Due Date:
The final project is due in its completed form (meeting all requirements set forth in this document and the approved project proposal) by the end of the Final Exam slot scheduled for the class: **2:30 PM on Friday 8/14**. As is the case with the milestones, the content will be pulled from the repository at the deadline time. Any content pushed to the repository late will not be considered unless prior arrangements have been made with the instructor to pull content at a different time. **To avoid any problems with files not being correctly pushed to the repository it is advised that the student should clone a second copy of the repository and test it for completeness.**

Milestones:
At the times listed below the instructor will check out the contents of the personal repository and review it against the requirement listed. Meeting the requirements listed for each milestone will be worth the specified percentage of the final grade regardless of the functionality of the final project at the end of the course.
<table>
<thead>
<tr>
<th>#</th>
<th>Milestone Requirement</th>
<th>Due Date</th>
<th>Grade %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A PDF and any supporting documents representing a proposal for the final project must be checked into a directory named Proposal at the root of the repository. The proposal must specify, at a minimum, the functionality that will meet the required criteria and at least 2 of the optional criteria; including descriptions of individual views and their functionality that take into consideration the components that are available as part of the SDK. Low-fidelity user interface mockups (white board drawing, sketches, etc) are encouraged.</td>
<td>12 Noon, 7/20</td>
<td>3%</td>
</tr>
<tr>
<td>2</td>
<td>A final revision of the proposal from milestone 1, incorporating any feedback received.</td>
<td>12 Noon, 7/24</td>
<td>3%</td>
</tr>
<tr>
<td>3</td>
<td>An initial version of the final project with at least half of the scenes from the project proposal in place and somewhat functional; their basic intent should be clear and implemented, and transitioning between them should also be implemented.</td>
<td>12 Noon, 7/31</td>
<td>3%</td>
</tr>
<tr>
<td>4</td>
<td>The project from milestone 3 with additional progress towards implementing the application proposed. At least three-quarters of the scenes from the project proposal should be in place and somewhat functional; their basic intent should be clear and implemented, and transitioning between them should also be implemented. An initial data model to support the application functionality should be in place and utilized by at least one part of the application</td>
<td>12 Noon, 8/7</td>
<td>3%</td>
</tr>
<tr>
<td>5</td>
<td>The project from milestone 4 with significant progress towards implementing the application proposed. All of the scenes from the project proposal should be in place and mostly functional. Transitions between the scenes should be implemented. The data model should contain all necessary entities and be utilized at least partially by all relevant scenes.</td>
<td>12 Noon, 8/12</td>
<td>3%</td>
</tr>
</tbody>
</table>

NOTE: The content will be pulled from the repository at the deadline time, so any additional content pushed after the deadline will not be reviewed unless prior arrangements have been made with the instructor to pull content at a different timestamp. If arrangements are not made and the content does not meet the requirement the student can request to have the missed milestone reviewed 24-hours later with a maximum grade of 80%.
Required Project Criteria:

The Obvious:
• Run without crashing
• No significant memory leaks (i.e., exhibits a stable heap size)
• Have a bug free and functioning user interface
• Be responsive to the user on reasonable hardware (no “long running” tasks on the main thread)
• Accomplish the goals of the application as put forth in these requirements and the approved project proposal

The Fundamental:
• Utilize at least one container view controller (navigation controller, tab bar controller, etc.)
• Be made up of at least 4 different non-trivial scenes that exist for a justifiable reason
• Support appropriate multi-tasking modes and system notifications
• Persist any user state, settings, and data between user sessions in an appropriate way (no plain text sensitive data!)
• Load user interface resources from Storyboard/NIB files as appropriate
• Have fully specified layout using Auto Layout
• Utilize version control during development
• Respect the MVC design pattern; appropriately separating concerns between the Modal, View, and Controller layers
• Persist user data using Core Data in a data model that contains at least 3 different entities that exist for a justifiable reason
• Provide simple user interface animation to give context for user actions as appropriate

Optional Project Criteria (Pick at Least Two):

The Interesting:
• Implement interesting gesture recognition to perform appropriate tasks and implement a custom view that performs non-trivial custom drawing
• Perform a complex task, which is relevant to the app, that would otherwise cause significant user interface lag by utilizing Grand Central Dispatch or a subclass of
• Allow the user to interact with the device camera and/or photos
• Allow the user to interact with the device address book and/or calendar events and/or send relevant emails
• Present data using an integrated map with appropriate annotations
• Play sounds and/or movies that are relevant to the application (i.e., the application should require media playback for its functionality - random button noises won’t count)
• Utilize the device motion sensors in a manner relevant to the functionality of the application (acceleration, location, rotation)
• Something interesting not listed here, with instructor approval