Name: Ethan Hunter
Title: cafe.connor.fun
Project Summary: cafe.connor.fun is a webapp for improving cafe discovery. Users will be able to use the app the find nearby cafes that they haven't yet visited. The app will be a game, awarding points to users when they visit cafes. The app will incentivize visiting cafes that the users hasn't been to yet.
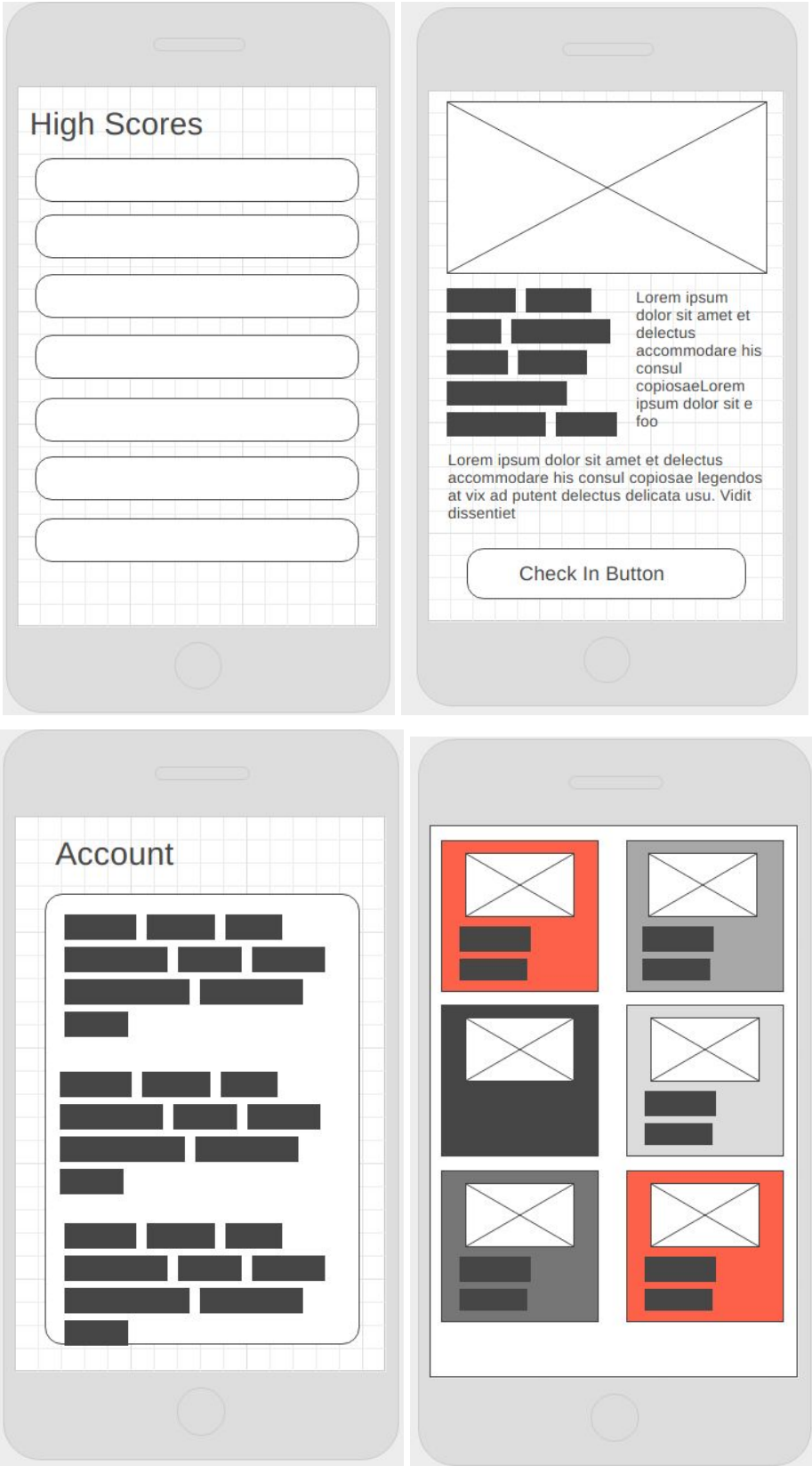
The project will also include the development of the Lighter Web Framework which provides a high-level abstraction for building RESTful web services. Lighter is an annotation based framework that focuses on fast service startup times, high throughput, and straightforward testing.

Actors: Users
Project Requirements

UC-ID	Description
UC-01	The user must be able to view cafes near to their current location
UC-02	Details of any cafe should be accessible to the user
UC-03	Cafes should be searchable - the search should prioritize by location
UC-04	User account details must be visible to each users. User should only be able to view their own account
UC-05	Users can check-in at a cafe to earn points
UC-06	Users can view the potential score of checking in at a cafe
UC-06	Users can view high-score tables that show the top scores among all users

UI Mockups

High Scores

Check In Button

Lorem ipsum dolor sit amet et delectus accommodare his consul copiosaeLorem ipsum dolor sit e foo

Lorem ipsum dolor sit amet et delectus accommodare his consul copiosae legends at vix ad putent delectus delicata usu. Vidit dissentiet
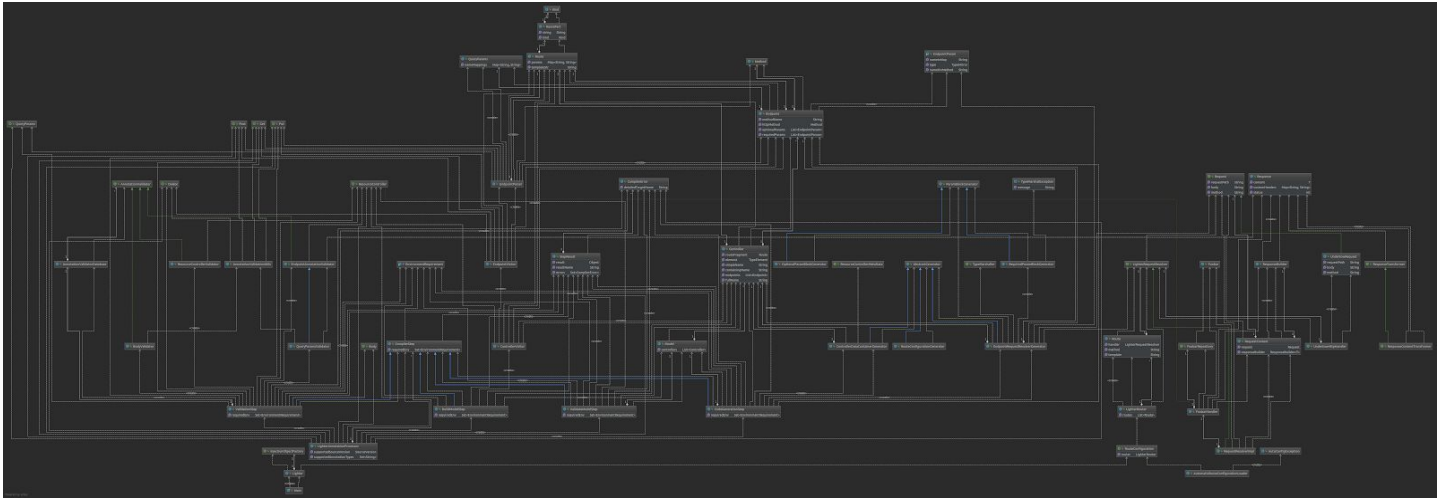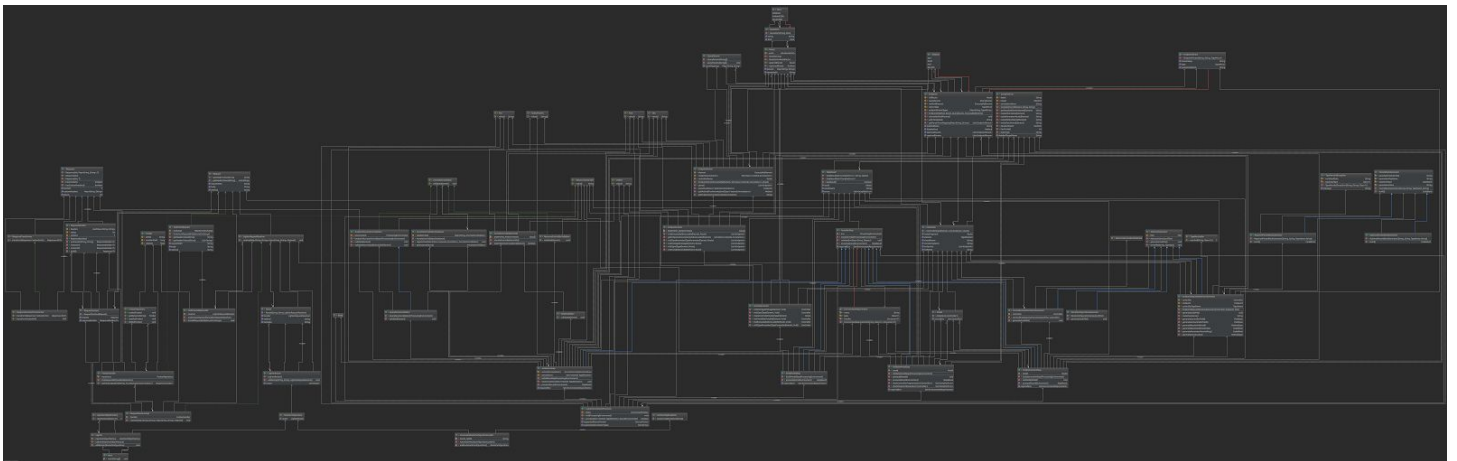
Account

Class Diagram

The project is broken into two parts. The first part is the Lighter Web Framework which provides a high-level abstraction for RESTful applications.

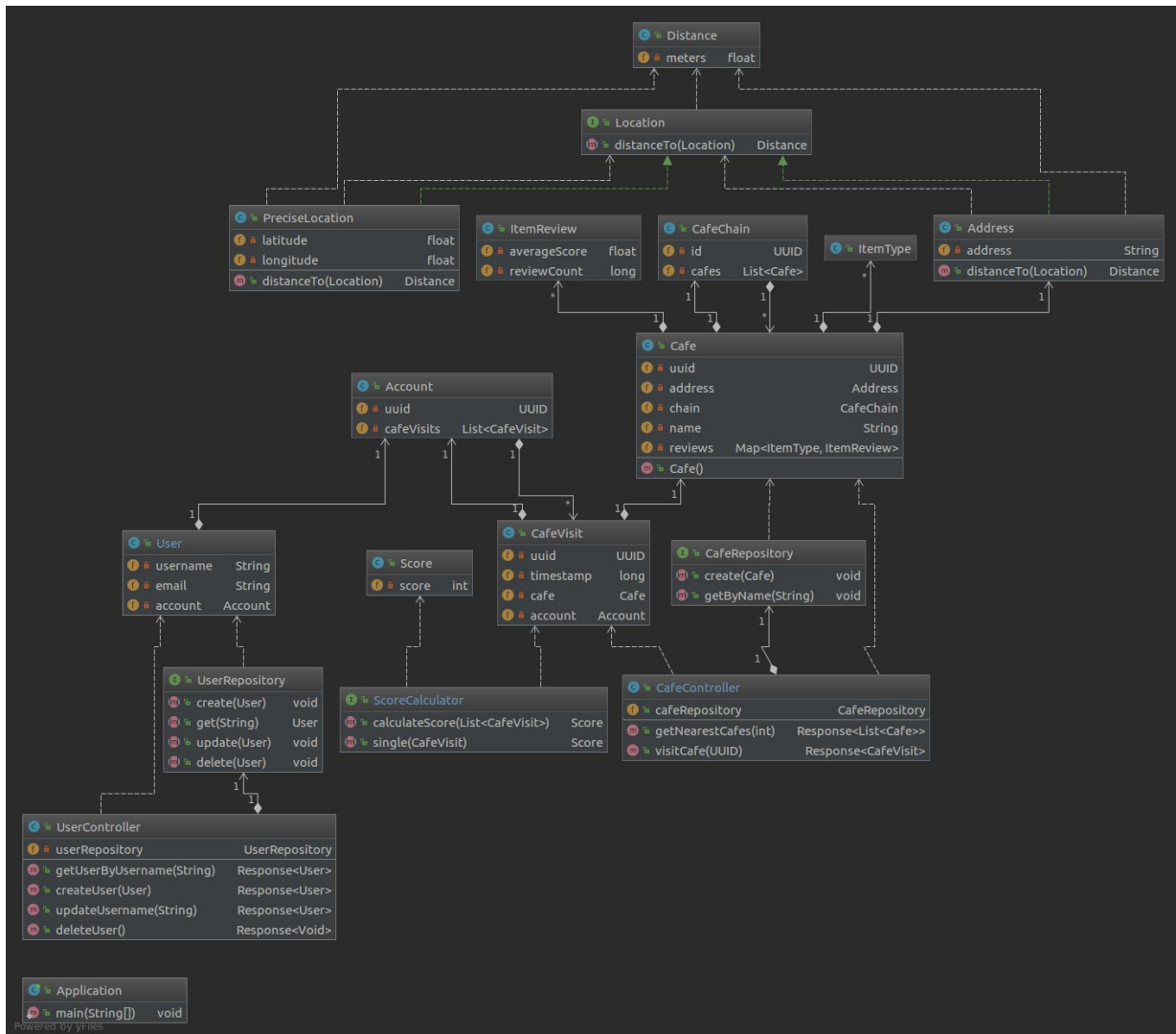This is a simplified diagram for Lighter (some attributes are hidden to improve readability)

This is the full diagram for Lighter:

(Both these images are very high resolution, so you should be able to zoom in to see better - Unfortunately even mildly complex systems are difficult to represent clearly using diagrams like this)

cafe.connor.fun is a simple webapp built using this framework (on next page):

Distance
🔘 meters   float

Location
distanceTo(Location)   Distance

PreciseLocation
latitude   float
longitude   float
distanceTo(Location)   Distance

ItemReview
averageScore   float
reviewCount   long

CafeChain
id   UUID
cafes   List<Cafe>

ItemType

Address
address   String
distanceTo(Location)   Distance

Cafe
uuid   UUID
address   Address
chain   CafeChain
name   String
reviews   Map<ItemType, ItemReview>
Cafe()

Account
uuid   UUID
cafeVisits   List<CafeVisit>

User
username   String
email   String
account   Account

Score
score   int

CafeVisit
uuid   UUID
timestamp   long
cafe   Cafe
account   Account

CafeRepository
create(Cafe)   void
getByName(String)   void

UserRepository
create(User)   void
get(String)   User
update(User)   void
delete(User)   void

ScoreCalculator
calculateScore(List<CafeVisit>)   Score
single(CafeVisit)   Score

CafeController
cafeRepository   CafeRepository
getNearestCafes(int)   Response<List<Cafe>>
visitCafe(UUID)   Response<CafeVisit>

UserController
userRepository   UserRepository
getUserByUsername(String)   Response<User>
createUser(User)   Response<User>
updateUsername(String)   Response<User>
deleteUser()   Response<Void>

Application
main(String[])   void

Powered by yFiles

Although "Application" appears to be a hanging class, it's not. Application contains the main method where the "Lighter" object is constructed. Since the rest of the application classes are auto-wired using the Lighter compile time application model builder, "Application" has no direct reference to them in the source.